

MAKE | BUILD | HACK | CREATE

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc

July 2022

Issue #56

MAKER'S MARKET

PLUS

• **Rocket Glider**
Like the space shuttle,
but smaller

• **CircuitPython P10**
Sending data quickly

Turn your **IDEAS**
into products that

SELL

**Alex
Glow**

Build electronics
without feeling
like a jerk



Jul. 2022
Issue #56 £6



56

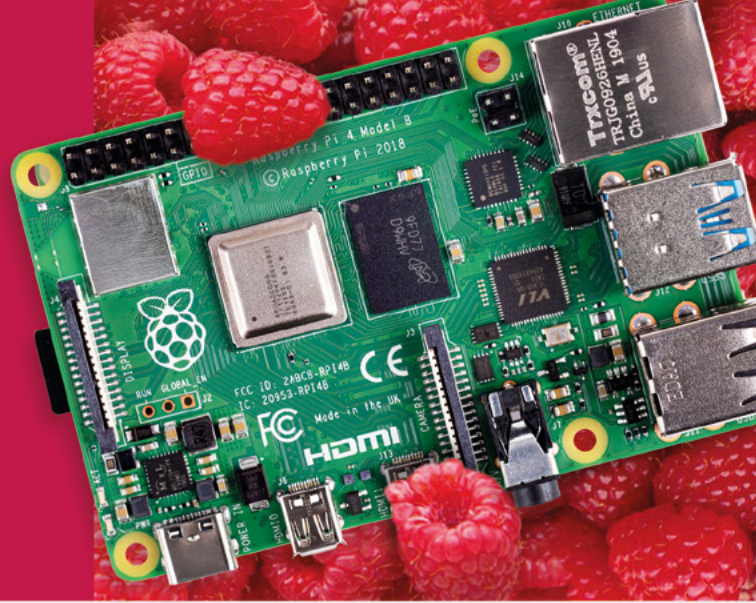


3D PRINTING POLYHEDRA SUBLIMATION FIREFLIES

VISIT...

LANZAROTE
Caliente.COM

American Raspberry Pi Shop



- Displays
- Cases
- Project Kits
- Add-on Boards
- HATs
- Arcade
- Cameras
- Cables and Connectors
- Sensors
- Swag
- Power Options
- GPIO and Prototyping

Partner and official reseller for top Pi brands:



and many
others!

Price, service, design,
and logistics support for
VOLUME PROJECTS

USA

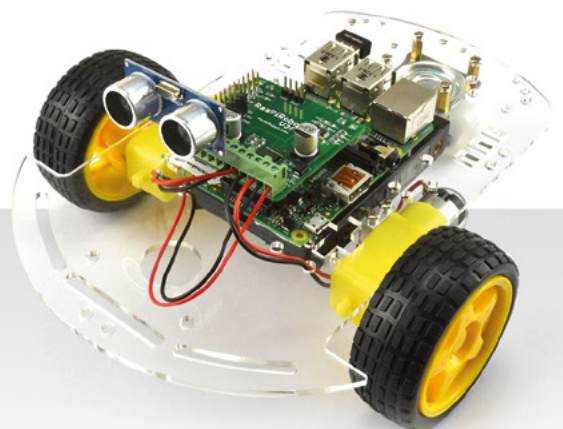


PiShop.us

Canada



PiShop.ca



Raspberry Pi

APPROVED RESELLER



Welcome to HackSpace magazine

Almost all of us have daydreamed about turning one of our projects into a product and selling it. The truth is that there's a lot more to making something that sells than just a good idea. It takes a good idea and a lot of hard work to productise it. This means making sure that it can be manufactured, tested, and properly packaged. And doing all that in a way that lets you hit a price point that people will pay.

The truth is that there's a lot more to making something that sells than just a good idea.
It takes a good idea and a lot of hard work to productise it

This might sound exhausting, but you are not alone. Generations of makers before you have managed to sell the things they've made. In this issue,

we're tapping into this wisdom to find out what you need to know about selling your wares.

If you're less commercially minded, we've also got some great content for you, whether you're looking to make colourful PCBs, update your laser cutter, or rocket towards space.

BEN EVERARD

Editor ben.everard@raspberrypi.com

Got a comment, question, or thought about HackSpace magazine?

get in touch at hsmag.cc/hello

GET IN TOUCH

hackspace@raspberrypi.com

[f hackspacemag](#)

[v hackspacemag](#)

ONLINE

hsmag.cc



PAGE **42**
FREE PICO
WHEN YOU
SUBSCRIBE

EDITORIAL

Editor

Ben Everard

ben.everard@raspberrypi.com

Features Editor

Andrew Gregory

andrew.gregory@raspberrypi.com

Sub-Editors

David Higgs, Nicola King

DESIGN

Critical Media

criticalmedia.co.uk

Head of Design

Lee Allen

Designers

Sam Ribbits, Olivia Mitchell, Ty Logan

Photography

Brian O'Halloran

CONTRIBUTORS

Jo Hinchliffe, Marc de Vinck, Andrew Lewis, Mike Bedford, Nicola King, Rosie Hattersley, Phil King, Rob Miles, Marc Scott

PUBLISHING

Publishing Director

Russell Barnes

russell@raspberrypi.com

Advertising

Charlie Milligan

charlotte.milligan@raspberrypi.com

DISTRIBUTION

Seymour Distribution Ltd

2 East Poultry Ave,

London EC1A 9PT

[+44 \(0\)207 429 4000](tel:+442074294000)

SUBSCRIPTIONS

Unit 6, The Enterprise Centre, Kelvin Lane, Manor Royal, Crawley, West Sussex, RH10 9PE

To subscribe

[01293 312189](tel:+441293312189)

hsmag.cc/subscribe

Subscription queries

hackspace@subscriptionhelpline.co.uk



This magazine is printed on paper sourced from sustainable forests. The printer operates an environmental management system which has been assessed as conforming to ISO 14001.

HackSpace magazine is published by Raspberry Pi Ltd, Maurice Wilkes Building, St John's Innovation Park, Cowley Road, Cambridge, CB4 0DS. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0). ISSN: 2515-5148.

Contents



102

06

SPARK

- 06 Top Projects**
Wonderful world, beautiful projects
- 18 Objet 3d'art**
Give this idea a big hand...
- 20 Meet the Maker: Ryan Downes**
Zen and the art of bicycle wheel construction
- 26 Letters**
Sympathy for a broken 3D print
- 28 Kickstarting**
Build a supercomputer from Arm boards

31

LENS

- 32 Create a product**
Go from prototype to world domination. It's easy!
- 44 How I Made: Tag jousts**
Design and 3D-print a game
- 50 Interview: Alex Glow**
How to make electronics without feeling like a jerk
- 58 Improviser's Toolbox Concrete**
The perfect outdoor build medium for this summer
- 62 In the workshop**
Adding colour with sublimation printing

Cover Feature

CREATE A PRODUCT

Turn your one-off project into something that people will want to buy

32

Tutorial

Build a swing-wing rocket glider



94

The ideal build for kids who played with Transformers

110



In the workshop

Sublimation printing



62

Liven up your PCBs' boring old solder mask

67

FORGE

68

SoM NeoPixels

Use dithering to control the brightness of LEDs

74

SoM Firefly

Mimic an insect's mating call with Raspberry Pi Pico

78

SoM Raspberry Pi

Read sensors on a Sense HAT with CDP Studio

82

Tutorial Polyhedrons

Use maths and 2D paper to make 3D shapes

88

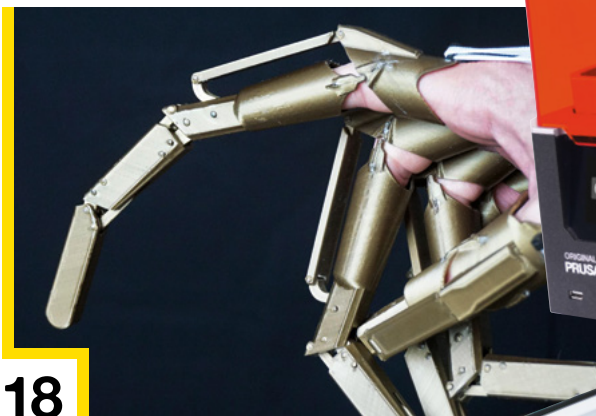
Tutorial Laser cutter – beyond the K40

We love the smell of burning wood in the morning

94

Tutorial Swing-wing rocket glider

Stick a rocket on a balsa wood glider – to the stars!



18

Interview

Alex Glow



50

Make better stuff, in a more sustainable way – here's how



112



78

88

101

FIELD TEST

102

Best of Breed

Musical diversions for makers

110

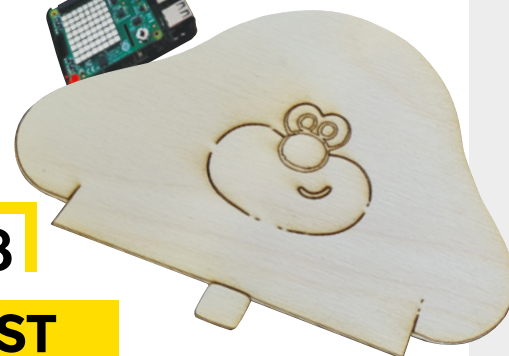
Review Phyx Space Shuttle soldering kit

Your very own PCB Space Shuttle Discovery

112

Review Prusa SL1S SPEED

All the detail of resin printing, with none of the hassle



Some of the tools and techniques shown in HackSpace Magazine are dangerous unless used with skill, experience and appropriate personal protection equipment. While we attempt to guide the reader, ultimately you are responsible for your own safety and understanding the limits of yourself and your equipment. HackSpace Magazine is intended for an adult audience and some projects may be dangerous for children. Raspberry Pi Ltd does not accept responsibility for any injuries, damage to equipment, or costs incurred from projects, tutorials or suggestions in HackSpace Magazine. Laws and regulations covering many of the topics in HackSpace Magazine are different between countries, and are always subject to change. You are responsible for understanding the requirements in your jurisdiction and ensuring that you comply with them. Some manufacturers place limits on the use of their hardware which some projects or suggestions in HackSpace Magazine may go beyond. It is your responsibility to understand the manufacturer's limits. HackSpace magazine is published monthly by Raspberry Pi Ltd, Maurice Wilkes Building, St John's Innovation Park, Cowley Road, Cambridge, CB4 0DS, United Kingdom. Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701, is the mailing agent for copies distributed in the US. Application to mail at Periodicals prices is pending at Williamsport, PA. Postmaster please send address changes to HackSpace magazine c/o Publishers Service Associates, 2406 Reach Road, Williamsport, PA, 17701.

Enigma display

By Hack Modular

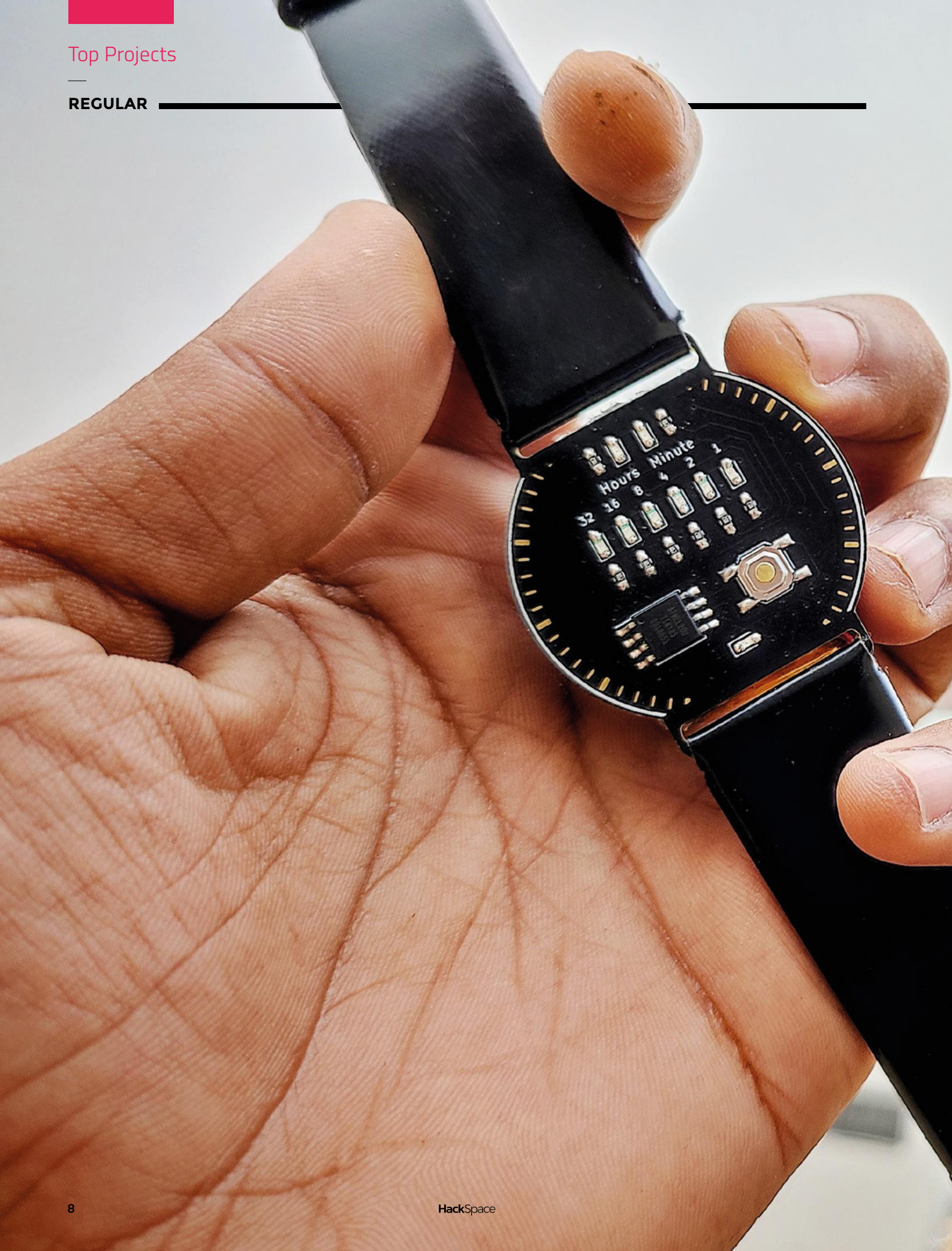
hsmag.cc/EnigmaLamp

This beautiful build looks like it's come from the bridge of a WW2 U-boat, but it's actually a completely new build. It's also a Eurorack module, albeit one that doesn't make any sound. It's a voltage-controlled character display, which means that when you plug it into your extensive modular synth setup and start playing the keyboard, different characters will light up when you play different notes. That's it! A brilliant, beautiful adornment, and an exploration of computing history. ▣

Right ▣
Each character
is assigned a
different semitone







BCD Watch (binary to decimal)

By [vishalsoniindia](#)

hsmag.cc/BinaryWatch

W

e love the aesthetic of surface-mount LEDs on a bare PCB, so we were fascinated when we found this watch. Even more so when we realised it uses binary-coded decimal – a system of counting that we're unfamiliar with. The watch is available to buy on Tindie,

or you can download the design files from the author's GitHub page and make your own. [□](#)

Left [□](#)

This is open hardware, so you can build your own BCD watch if you want to learn more

Solar robot

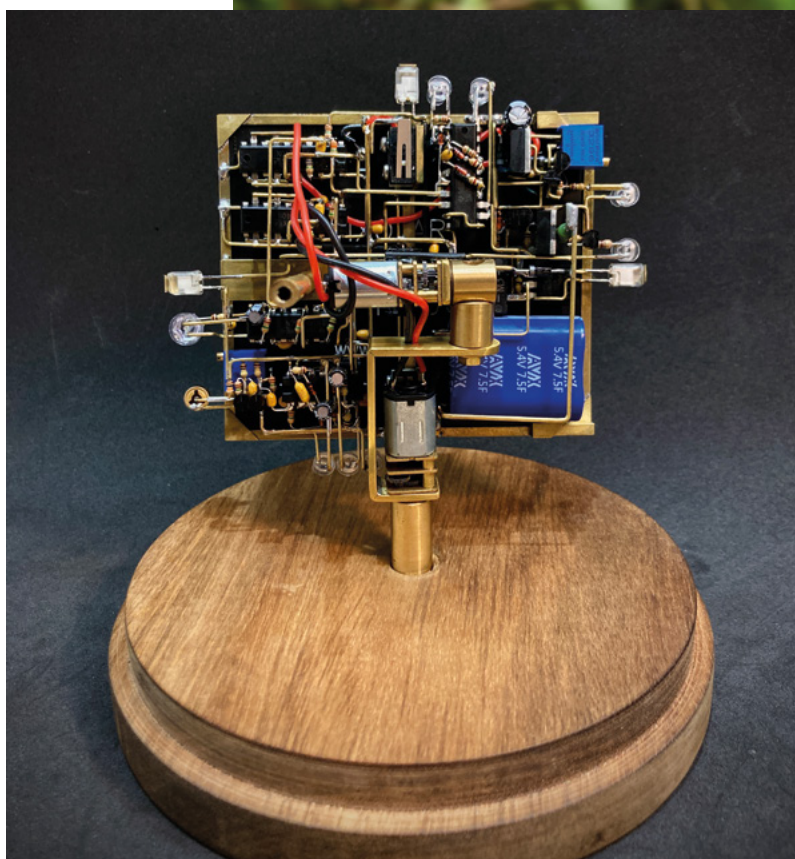
By Jack Spiggle

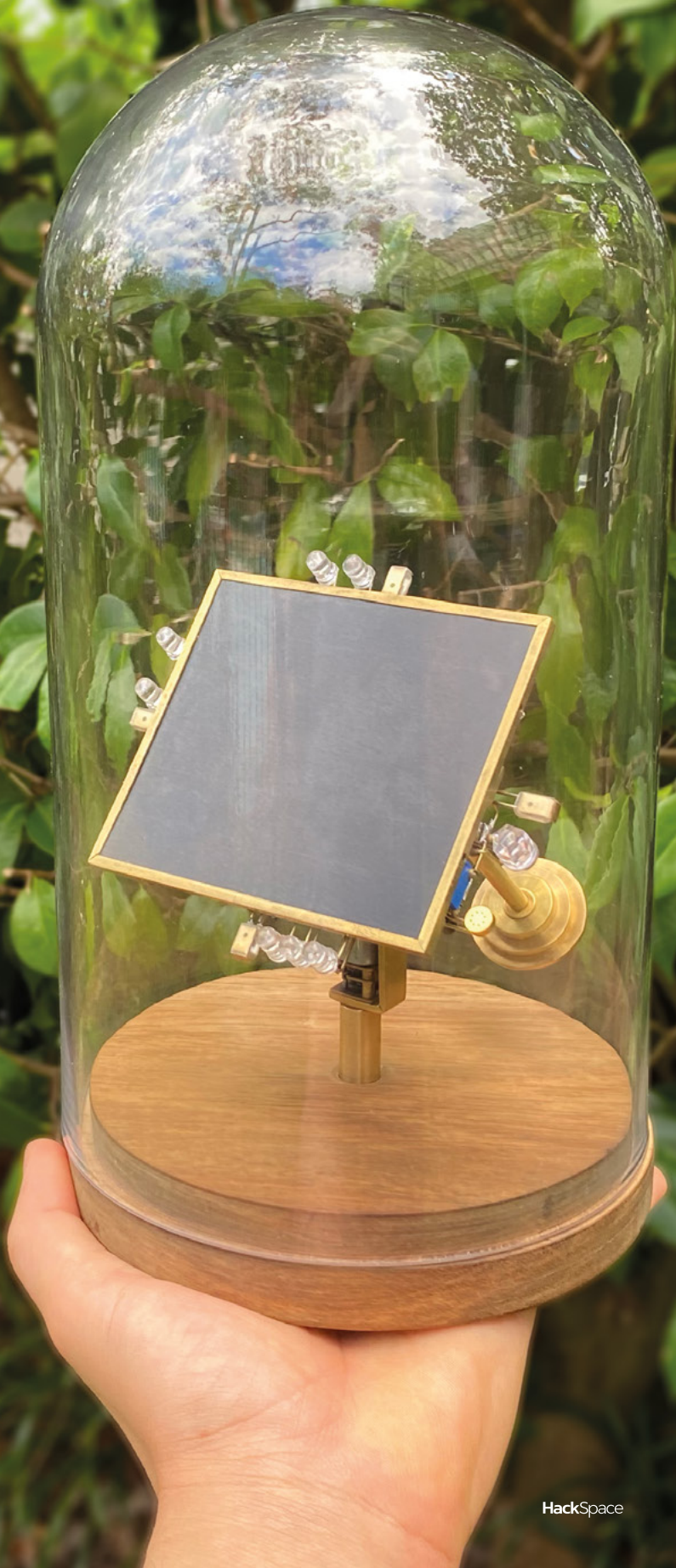
hsmag.cc/RobotHeadInAJar

We've featured the work of Jack Spiggle, aka NanoRobotGeek, a couple of times in past issues. And we're very proud to be able to say that he'll be writing a How I Made for our next issue, unveiling some of the thought processes behind this free-form, solar-powered kinetic sculpture. Like a lot of his work, this uses solar power stored in capacitors, which triggers an action when a certain amount of charge is reached. And like all of his work, it's absolutely beautiful. ▣

Right ▣

Brass is tricky to solder, but the finished work is so pleasing to behold





Welding helmet

By Andrew Lewis

 lewis-workshop.com

Here's another preview from the next issue: an enhanced welding helmet by Andrew Lewis. While it's possible to spend loads of money on a top-of-the-range welding helmet, for most weekend welders, it's just not worth it. If you spend more time grinding off messy welds than you do actually welding, there's no way you're going to spend money on luxuries like a fan – but you might follow Andrew's example and enhance a lower-end model. Look out for step-by-step instructions in issue 57. 

Right

Look the part while you're sticking hot bits of metal together





Loopie lamp

By CodersCafeTech

hsmag.cc/LoopieLamp

The most frustrating thing in woodworking is things that are supposed to be square but aren't square.

Give us a nice right angle any day.

Our mass-produced world likes order, tessellation, and right angles. For ease of design, construction, and transportation, right angles make sense. So why has the maker of this WS2812B RGB LED-powered lamp chosen the way of the loop for this design?

Refreshingly, the answer is that it's better for the user. Rounded shapes are more common in nature, and so the theory is that they're easier on the eye and the brain. The circle also represents eternity, as it has no beginning and no end. Either way, this is an impressive build, achieved with simple tools, simple electronics, and a few bits of MDF. ▣

Right ▣

The light transitions are powered by a Digispark ATtiny85-based board



HackSpace

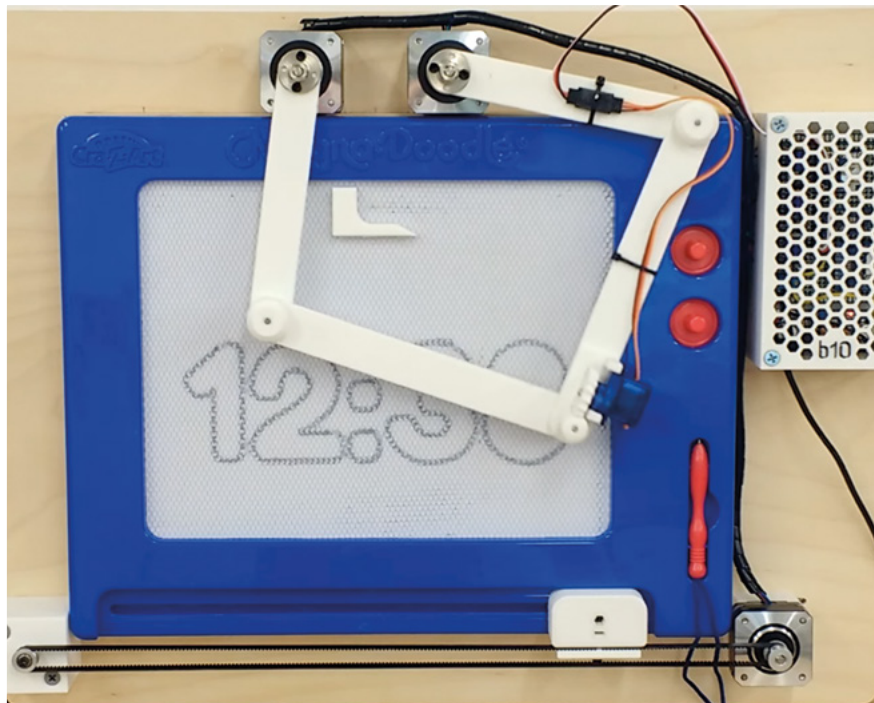


Magna Doodle Clock

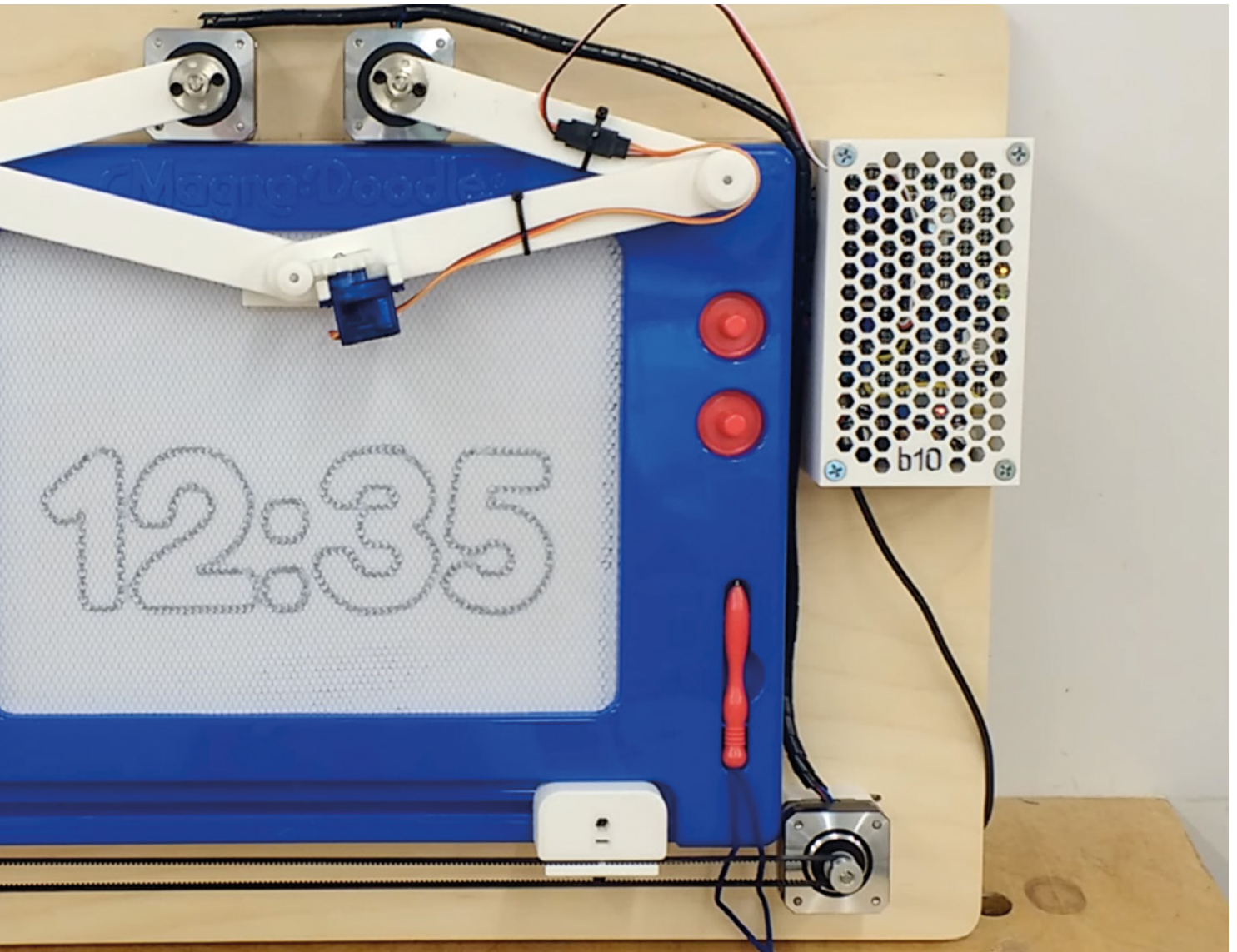
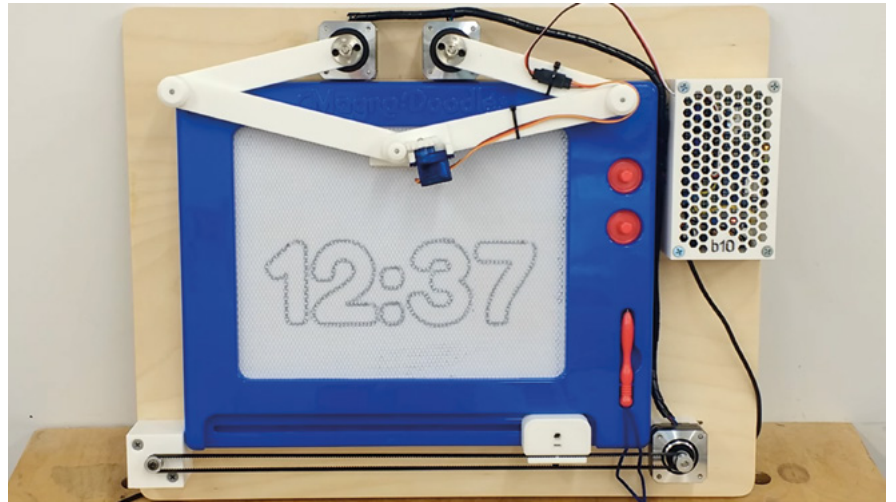
By MrRoboto19

hsmag.cc/MagnaDoodleClock

If you're sick of getting information from glowing screens, this project is for you. This brilliant clock draws the time on a Magna Doodle, then wipes the digits that have changed, and draws the time again, over and over. It's powered by an Arduino Nano, three stepper motors, and various other electronics – we want one for the office. Every hour it'll change to a random font stored on an SD card – the maker advises using fonts that don't overlap, as the machine needs to be able to wipe off the digits when the time updates. ▣



Right ▣
Classic magnetic
fun updated for
the maker age



Objet 3d'art

3D-printed artwork to bring more beauty into your life

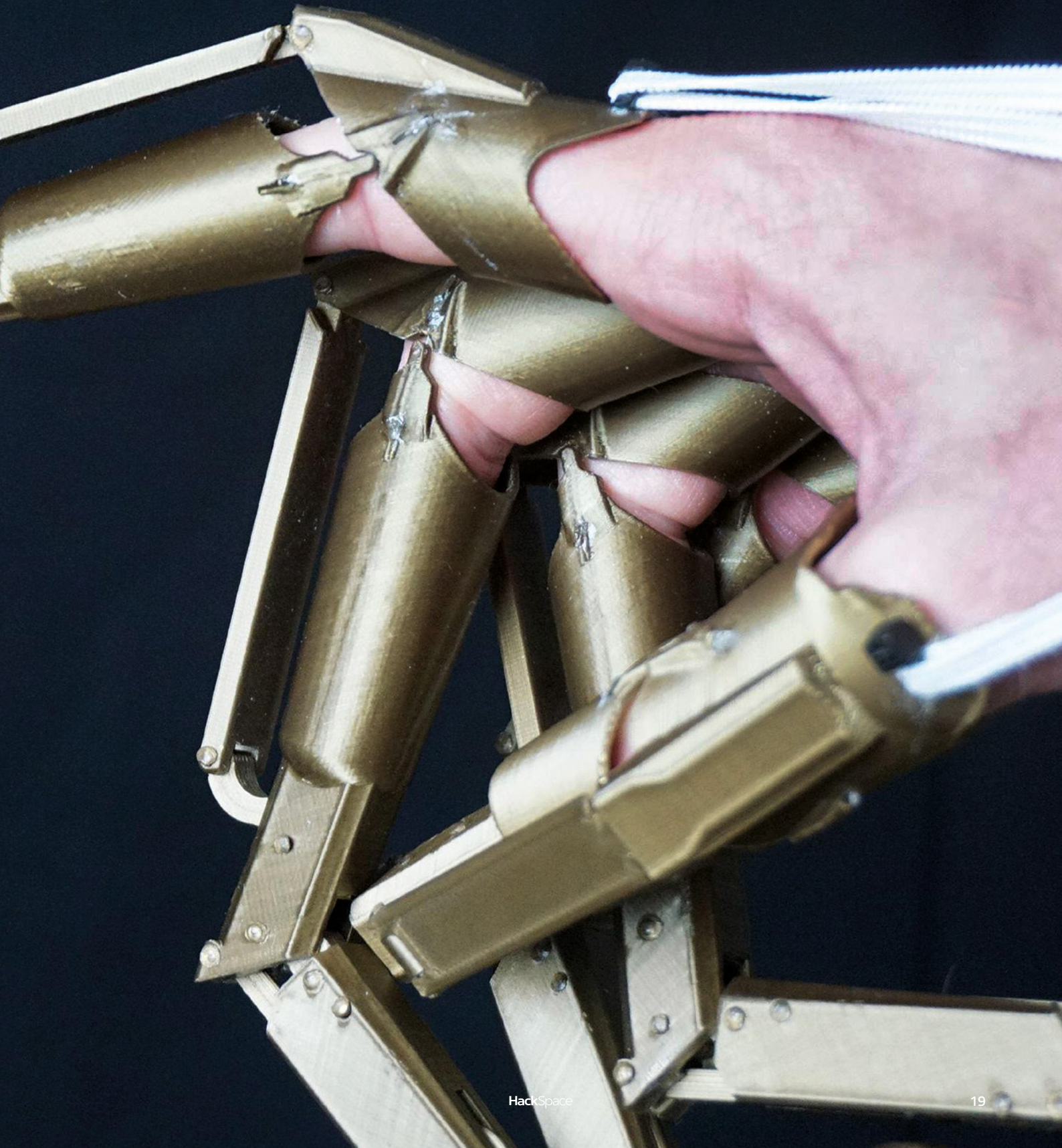
There's a phenomenon called the 'uncanny valley' wherein something that's almost – but not quite – human unsettles us far more than something that's

totally alien. That's what we thought when we saw this set of 3D-printed finger extensions by Instructables user Dragonator.

The creator says that, "The basic idea of the finger extensions is that it takes the movement of your finger, and extends it to several more joints." That much is obvious looking at it, but it's still weird. If you want to make a set of these for yourself, you'll need a 3D printer (preferably with some 1.75 mm filament), a couple of drill bits, wire... and that's about it. Get making now and you'll have the guts of a superb Halloween costume. ▣

➔ hsmag.cc/3DPrintedFingers





Meet The Maker: Ryan Downes

Zen and the art of building wheels

Pedal bicycles are one of the most personal machines. They are one of the few machines that are still widely used that are powered by the operator. You don't just provide the instructions for where it should go, you provide the power to send it there – this creates an intimate connection between rider and machine.

We spoke with Ryan Downes of RyanBuildsWheels about what it takes to turn lifeless bits of metal into machines that bring joy, and how he learned the dark art – or is it science – of wheel building. Here's what he had to say.

"I picked up a cheapish bike at university just to get to and from uni readily – I could get a few more minutes of lie-in in the mornings.

// You don't just provide the instructions for where it should go, **you provide the power to send it there**

//

"I'd got a few quid for my birthday from family. Enough for a not particularly great bike from Halfords. I went to pick it up but wasn't aware that bicycles come in boxes and need assembling. I was ready to ride my bike away.

"It was deep winter and I lived ten miles outside of Worcester, and of course, [Halfords] were like, 'It's in a box; it needs assembling. Can you come back tomorrow, or at least later on today?' I was like, 'No. I just spent what little money I had >

Right ♦
Ryan builds wheels by hand in his workshop in St Werburgh's







on a bus over here, and it's snowing – just give me the bike'.

"I took it home and found it came with a tool-kit. I thought, 'I think I can do this; I'm fairly practical'. I had to go and buy a few more tools, and I had to go and buy *The Complete Idiot's Guide to Bike Maintenance and Repair* and do some research, but I built it.

"The freedom that that machine gave me just blew my mind. All of a sudden, it wasn't just to save ten minutes getting to university in the morning; it was 'I can go into town on this', 'I can ride over to mates' places', and, 'I don't have to get taxis'. It was unreal. As luck would have it, another friend on the open mic scene had just recovered from 15 years of extremely hefty social anxiety, and he'd just started coming out into the world again. He'd had this really nice ten-speed racing bike built 15 years ago, but decided he didn't want it anymore. He was massive as well – he was six-foot-four.

"He asked one mate if he wanted it, and he rode that around for a bit and decided it wasn't for him. That person then asked me if I wanted it – I think I paid about 50 quid for it.

"All of a sudden, my little Halfords-bike-mind was blown. Even though I had to have the saddle all the way down, I was like, 'Man, these things are fast! These things are fun!'. I thought, I guess how far I'll ride.

"I found myself riding into the Welsh Borders, past Hereford from Worcester, and just loving the freedom that it bought. I fell in love with the romance of road cycling and the older Tour de France – from the era of the bike I was riding.

"I became obsessed with riding it, and changing things around on it. The wheels got buckled once – I was already invested in the politics of DIY. I didn't want to go to a bike shop; I didn't think there were any good bike shops [where I lived]. I had a look into how you make wheels straight, got the tools, and set to work. That was great fun.

"After this, I came to Bristol. I was getting more and more into bikes. I was volunteering at an excellent place called The Bristol Bike Project, which essentially fixes up old bikes that are donated to them. I really cut my teeth there.

"I thought I was an academic, and to an extent, I still am. I enjoy certain types of study – but I realised that what makes me most happy is tinkering. Using my hands and brain and making stuff happen – and understanding how stuff works. I wanted to get into bikes, and thought I wanted to be a bike mechanic. I was working as a courier here and there. I had worked as a courier in Brisbane for a while whilst also volunteering for a couple of projects [in Australia].

"I managed to talk my way into a little punky shop here in Bristol, up in Clifton. It was – in the worst sense of the word – a hacky shop. I won't name it, but back in the day, we were pretty slack, those of us that worked there. We'd go out for pints at lunchtime and go back to the workshop and be like, 'what was I doing?'.

GETTING BUSY

"That was only a part-time gig. I found myself in a situation where I really needed a full time job. I'd been squatting and doing other things outside of that, so I could work part-time and do the things I wanted to do. But the situation forced my hand, and I had to work full time. With enthusiasm, I talked my way into a bike shop that was probably one of the busiest and most commercial shops. I already thought I was a top dog because I'd worked in a city-centre shop, but as soon as I started working in this shop that was much busier and required you to really be on it, I realised how little I knew.

"I wanted to be really good. One of my favourite books is *Zen and the Art of Motorcycle Maintenance* by Robert M Pirsig. You should read it should you so desire – it's very much my bible. I wanted to approach the task in hand with 'Quality' – Pirsig talks about Quality with a capital Q as some sort of metaphysical thing that is inherent in the world. You're not actually separate from the object that you're working on. Too often, there's this dualistic subject-object thing when we come to working on →

Above ♦
Spokes are cut to length so that they've always got the perfect size in stock



things – bikes, etc. I am the mechanic; this is the bike. I do the thing; it behaves like this. Sure, it does, but actually there's much more of an emotional investment in it. It's not subject-object dualism. All of a sudden, it's what mood is the subject in? How much of a good time am I going to have fixing it? If it's just an annoying machine that won't work, of course it won't work! You and your readers are probably more than aware of this sort of stuff.

"I knew I had to get better and approach it with a degree of mindfulness. Mechanics is something that I'm a little bit obsessed with. Because I worked under such a great head mechanic – he went out on his own and now runs a great place called Robin's Cycle Services in Bristol – he was a great teacher. He'd never be angry at you if something went wrong. Instead, bless him, he'd be disappointed. He was a real father figure in that workshop. You didn't want to disappoint because everyone was in it together. That really made me want to push myself. I did, and eventually, the short version of the story after that is a close friend, who I worked with at Sustrans [a sustainable transport charity], wanted to set up a cycle café – Roll For The Soul. I got headhunted for that. It was nice to be trusted to be the person responsible for that. It was

also nice to have my own workspace and to really be able to figure out how and why I wanted to do bikes.

"The website's still up, and so is one of the blogs – I think I entitled it *Why we wrench* – which talks about the metaphysics of quality.

"I'd started building wheels back when I was squatting. The way that wheels work began to interest me more and more. In shops, I'd already become the person in Bristol to speak to if you wanted to build wheels. The more I got into it, the more I realised that, in the industry, we're not giving wheels the attention that we should. There are a

few key concepts to understand around wheel building.

"People say that wheel building is a dark art, but actually, it's basic Newtonian physics, and it's a science. Anyone can build wheels with a half-decent

understanding of how they work. I keep saying that they are a prestressed tension balanced structure, much like the Clifton Suspension Bridge.

"I started more of a deep dive into how and why wheels work, and I realised that there was much more to wheels than I'd been taught, and I realised that I loved it. It was a really relaxing, meditative flow-like state to find yourself in. There is a certain amount of, if not creativity, then creation. You start

// I'd already become the person in Bristol to speak to if you wanted to build wheels //



Right ♦
New bikes waiting
for new wheels

with a bunch of components. You spec it, and hey presto, you have a thing that you've made. It was a real nice way to spend your time.

"I think that obsession started to show while working in Roll For The Soul. I suppose I got less excited offering people advice on how to fix their lower-quality machines. One of the things we never did in the shop I ran was offer bikes for sale. That was a very conscious decision. We wanted to be able to say to people, very honestly, 'the machine you've given us is not financially viable to get this into a state it can be ridden. You haven't cared for it; you've ridden it into the ground, and it's dead. If you're going to use it this much, these things need caring for.'

"It's hard for some people to understand that if you want a bike these days, you need to spend 350 or 400 quid. That's the entry level for an average-quality, new commuting machine.

"My own desire to be wrenching on stuff, and things like wheel building, decreased, and I wasn't getting as much out of it.

"I started building wheels one day a week working out of the shop – one day a week became three days a week, which became six days a week, and before I knew it, I just couldn't do it any more – it was time to go full time. Me and my friend Pi (who runs Clandestine) – he's a frame builder here in Bristol – decided to set up shop together. It was a little maker space in Lawrence Hill above some music studios.

"I started slow – there was no business plan. These days I do have a business plan if I do something. I need to know that something is going to work if I'm going to change tack. People should look out for a YouTube channel that's being started soon. It'll have lots of information on DIY stuff. I want to get out the idea that wheel building is something you can do in your own home if you feel adept or just want to have a go. With some basic tools, you can make some good wheels. The tolerances we have will be higher than people can do at home, but I want to crack that open and explore the magic of spoked structures.

SPOKE EASY

"Tolerances kept for spoke tension are much higher, and that's one of the main things that's going to affect spoke life. Anyone who's building wheels by hand and for a living wants to stand by the quality of their product, so with very few exceptions, no one's going to go skimping on rims or things like that. A lot of [factory-made] wheels are a race to the bottom in terms of margin using spokes that might not last as long.



WHY GET HANDMADE WHEELS?

"The most common place for manufacturers to save money is on the hubs. It's also one of the easiest places to save weight. With wheels, what you get is this lightweight headline, but talking about wheels purely in terms of their weight is nonsensical. You've got to consider how the whole structure works. It's a rotating mass, so you can save weight by spec'ing a cheaper hub with less material really easily, but that hub just falls apart. You want to save weight externally, but you don't want it to be too light.

"When you buy a custom-made wheel, you get a wheel that's spec'ed for you. Some people need more or fewer spokes, or a thicker gauge of spokes. Some people need to know that their hub axle is really sturdy because they're doing a lot of touring. The possibilities are endless. It might be as simple as you want the nipples on your wheels to be red because your bike needs a bit of bling.

"One of my favourite tasks is building wheels for more powerful riders. That's really nice to be able to democratise wheels and cycling for more people and allow a wider variety of people to ride bikes. The nice thing is, amongst all the builders that I know, go to whoever you dig the most. Either because they're local to you or they have politics you agree with. They might have aesthetics you like or have a cool logo. You might just enjoy reading their website. Some focus more on performance cycling, and you might be keen to get the most out of your gear. Go to whoever intrigues you the most. I know for certain that all of my peers are building wheels as good as me. If you go to someone down the road, you're going to get a great wheel". □

Above ♦
Checking the tension on the newly-fitted spokes

Letters

ATTENTION ALL MAKERS!

If you have something you'd like to get off your chest (or even throw a word of praise in our direction), let us know at hsmag.cc/hello

PLOTTING

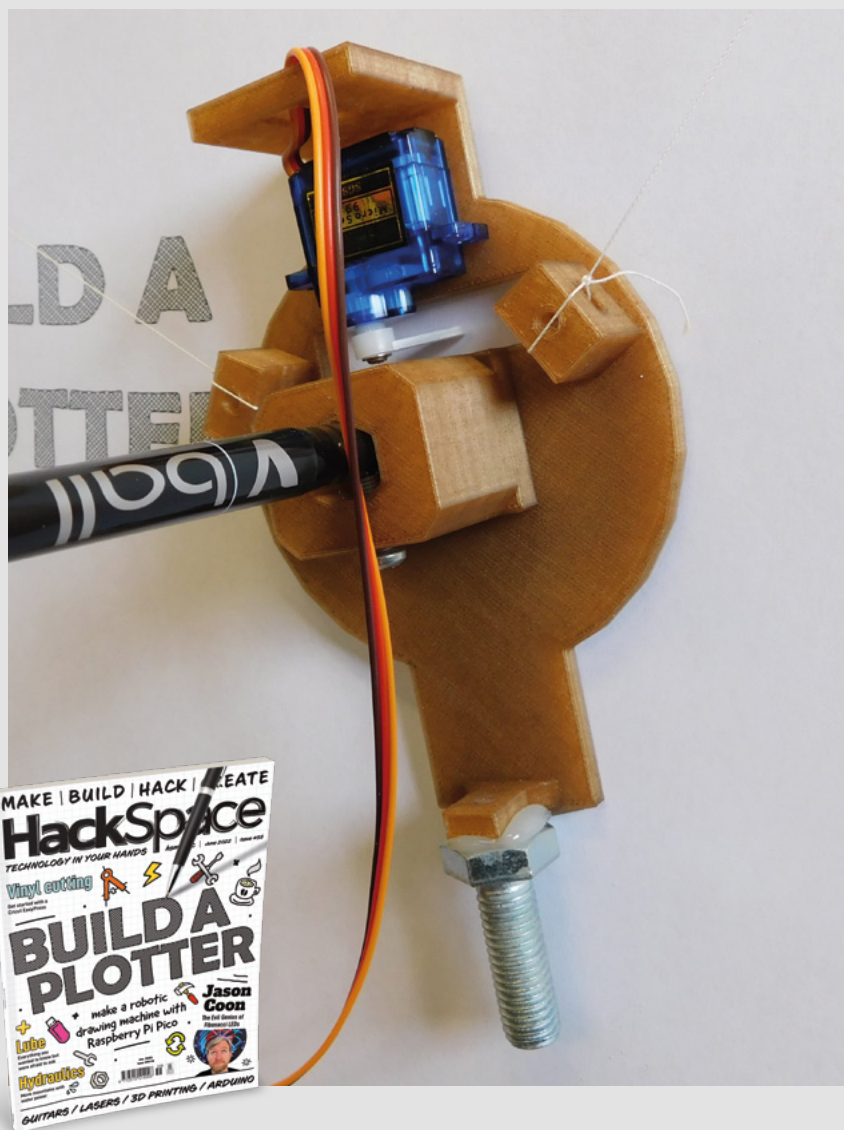
I was drawn (pun definitely intended) in by something you wrote last issue about the 'build a plotter' cover feature: "Nobody needs a plotter – it won't solve a problem you have, but it is just fun."

I find myself repeating, more or less verbatim, this argument whenever my wife asks why I need another development board, or another woodworking plane, or another power tool. It's pointless, but that is the point. I'm just amusing myself, and if I manage to make anything useful, that's a nice side effect.

Dean

Melbourne

Ben says: Making stuff just for fun is the be-all and end-all in my book. It's like growing your own food on an allotment: it's never going to be as efficient to grow a calorie of food yourself as it is to let someone else do it and bring it to your local supermarket. But that's only one side of the coin. You've also got the sheer joy of being out in the open air, of getting out of the house, watching things grow under your loving eye. Making is very similar, in that I know I'm not saving any money; in many cases I'm not even making anything useful. But what I am doing is finding a flow state, concentrating on something that isn't work, and satisfying the human urge to create. Building that plotter was very silly, but it was also a lot of fun. For that, it was well worth it.





ZERO TO HERO

The broken 3D-printed guitar that you used turned into a games controller: bad luck! I can see the logic of what you did, as can anyone looking at the image. If you'd asked me to 3D-print a guitar neck, I'd probably have done the same thing, with each layer built up from the print bed making the guitar neck longer. It's just the instinctive way you'd do it if you were familiar with 3D printing in any capacity. But, if I look at my own guitars, which are made the traditional way out of wood, I can see quite clearly that the layer lines put down by the tree run lengthwise. Even though there's one in the same room as my 3D printer, I would have made the same mistake, because that's the way that the medium naturally takes you. I guess the world only lets you shape it the way it wants to be shaped – either way, good work on turning a failed print into something useful!

Lucy
Bedfordshire

Ben says: I understand what you say about there being a 'natural' way to think of 3D-printed objects, but that isn't what happened with the guitar neck. I thought long and hard about it, considered all my options, and decided with a clear head to do it the wrong way. Never mind. Like NASA, HackSpace magazine has a no-blame culture. This is how we learn from our mistakes...

LIVE, LAUGH, LUBE

Like most people who have entered the doors of a maker space, I've seen the engineering flow chart (if it moves and it's not supposed to move, apply duct tape; if it doesn't move and it's supposed to move, apply WD-40). What I'd not realised though was how much there is to greases and oils. I know the information is all out there already in books and on the internet, but thanks for putting it into my hands last issue.

Brian
Stoke

Ben says: The skill of editing a magazine is bringing pertinent stuff together in one place. At least, that's what we're told. We just let the brilliant Andrew Lewis get on with things, as he is doing this issue with the latest part of his deep dive into what makes a laser cutter work. Turn to page 88 and enrich your mind with his words.



CROWDFUNDING NOW

Mixtile Blade 3

The Arm processor board that likes to play with friends

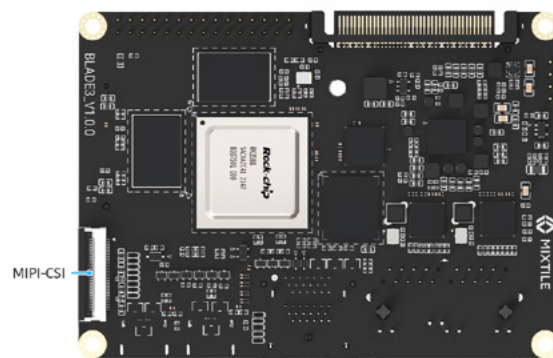
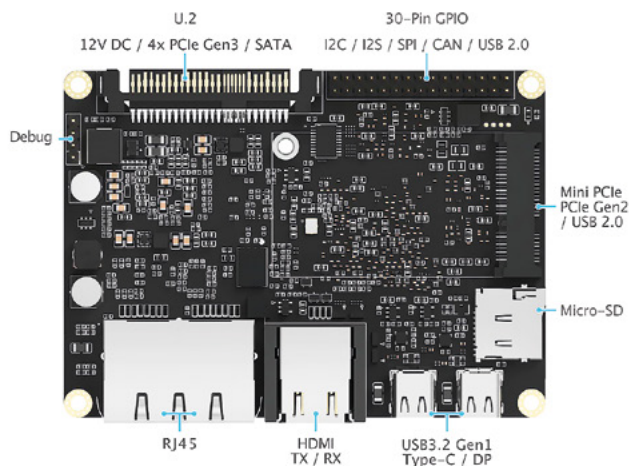
From \$169 | hsmag.cc/mixtile | Delivery: Oct 2022

The Mixtile Blade 3 packs a Rockchip Octa-core Cortex-A76/A55 processor running at up to 2.4GHz, with up to 32GB of memory, and 256GB of eMMC storage. The real tick, though, isn't the processor, but the form-factor. It's stackable with a PCI Gen 3 edge connector. This means that it's really easy to link multiple Mixtile Blade 3s together to make a cluster.

There are a lot of trade-offs to consider when building computing clusters, but often the key decision comes down to whether or not you should have a few high-power processors, or lots of

low-power processors. While the Cortex-A76s at the heart of the Blade 3s are powerful by Arm standards, compared to most server-class x86 processors, they're relative lightweights. That does bring advantages, particularly with energy efficiency. Whether or not this is a good trade-off depends a lot on your particular processing tasks.

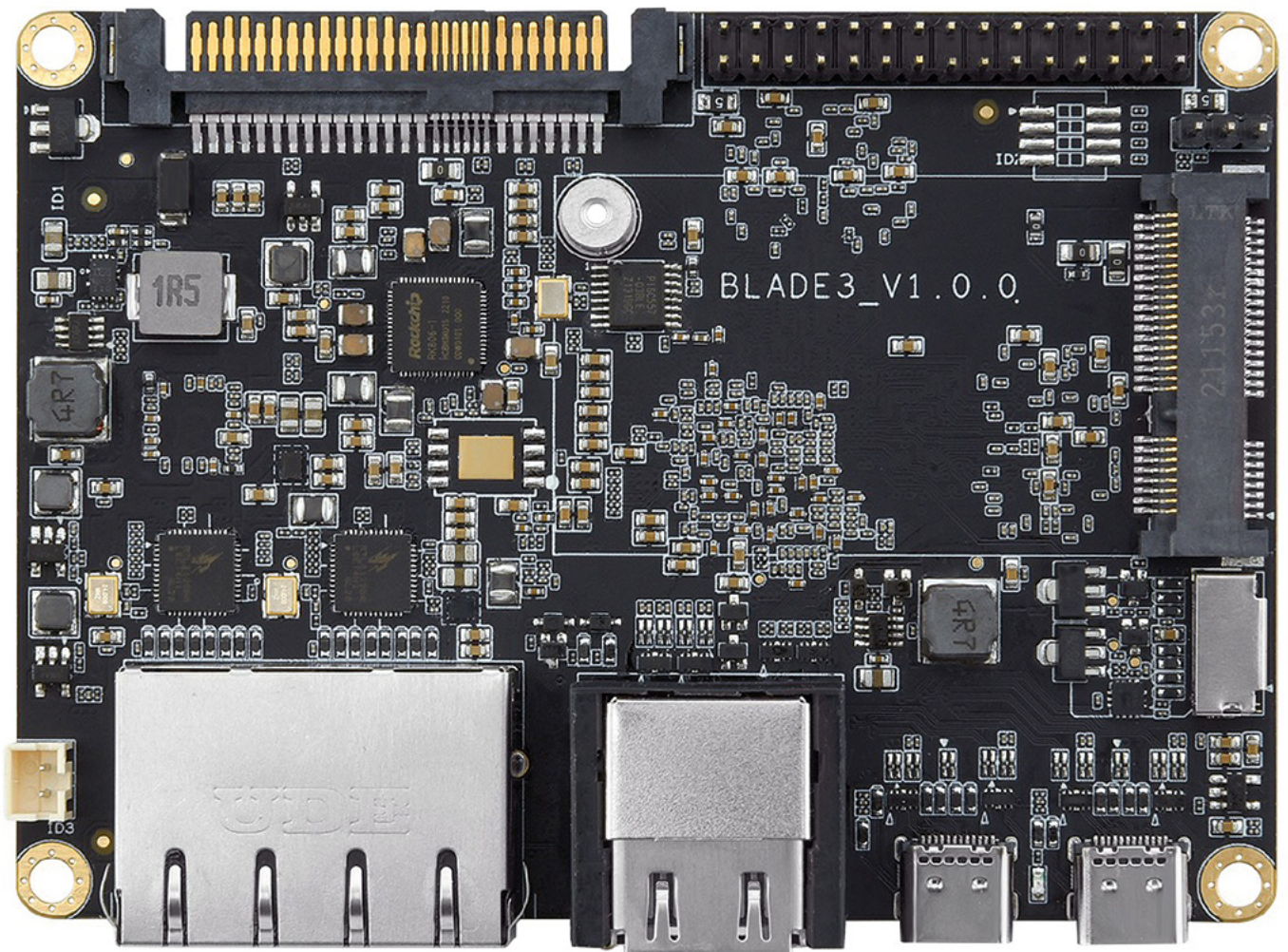
There's also a 40-pin GPIO header, allowing you to integrate a huge range of hardware with your machine, dual Gigabit Ethernet, and SATA 3.0. While we've not had our hands on this to test, it looks like it's a great option as a fully-featured Arm processor board. ■



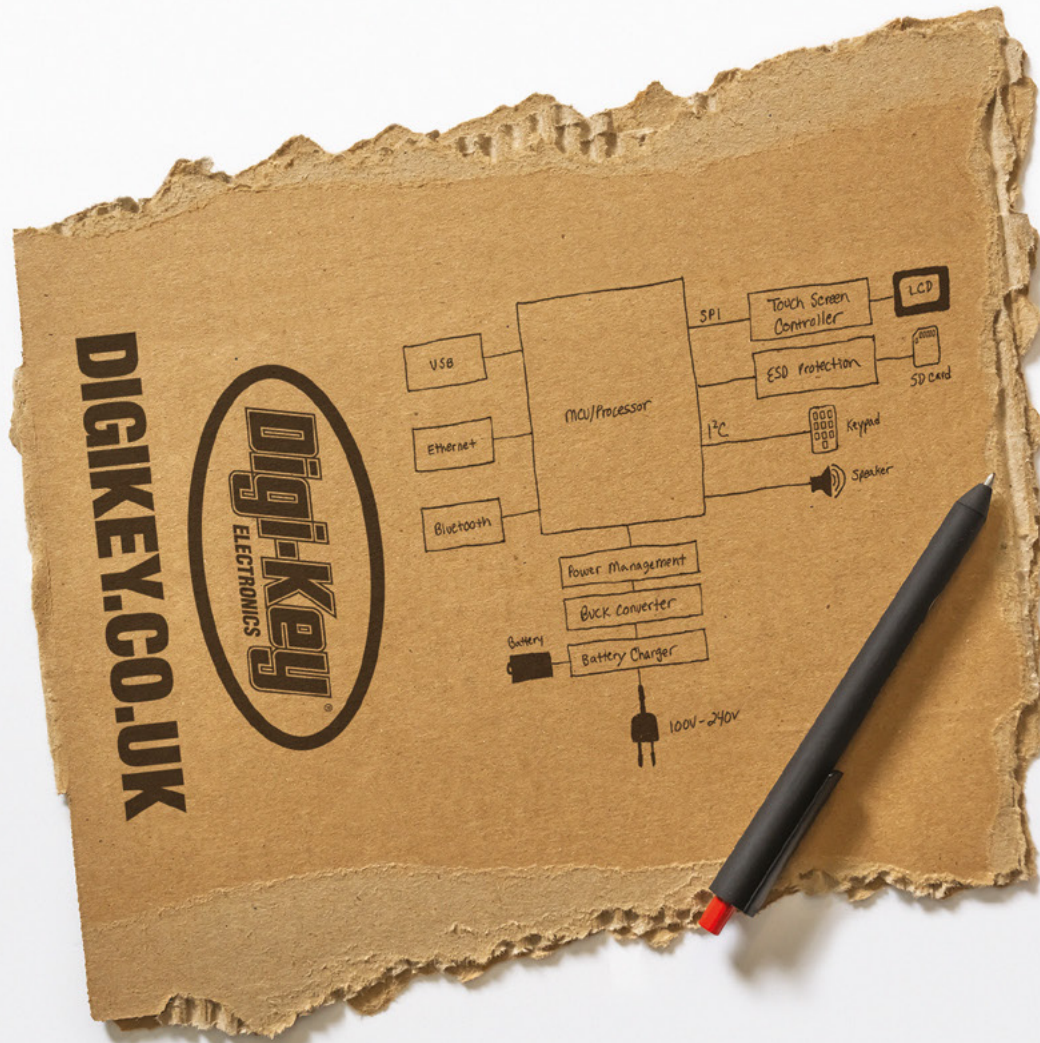
BUYER BEWARE

When backing a crowdfunding campaign, you are not purchasing a finished product, but supporting a project working on something new. There is a very real chance that the product will never ship and you'll lose your money. It's a great way to support projects you like and get some cheap hardware in the process, but if you use it purely as a chance to snag cheap stuff, you may find that you get burned.

Below 
The building
block of an
Arm cluster



IDEAS START HERE



From millions of in-stock parts to the latest new product inventory, we've got everything you need to turn breakthrough ideas into reality.

Get inspired at [digikey.co.uk](https://www.digikey.co.uk) or call 0800 587 0991.



Digi-Key is a franchised distributor for all supplier partners. New products added daily. Digi-Key and Digi-Key Electronics are registered trademarks of Digi-Key Electronics in the U.S. and other countries. © 2022 Digi-Key Electronics, 701 Brooks Ave. South, Thief River Falls, MN 56701, USA

 **ECIA MEMBER**
Supporting The Authorized Channel

LENS

HACK | MAKE | BUILD | CREATE

Uncover the technology that's powering the future

PG
44

HOW I MADE: TAG JOUSTS

Combine 3D printing and FreeCAD to make a game of skill, bluff, and chance



PG
50



INTERVIEW: ALEX GLOW

How to keep building electronics without leaving the planet spoiled for future generations

PG
58

IMPROVISER'S TOOLBOX: CONCRETE

The Romans used it in their weekend project builds – now you can too

PG
32

CREATE A PRODUCT

Take your idea from pipe dream to prototype and beyond

PG
62

IN THE WORKSHOP: SUBLIMATION PRINTING



We tried printing onto non-printable things. Success level: mixed

CREATE A PRODUCT

Thinking about turning your electronics project into an actual product? It's not easy, but you'll be creating something unique and learning a lot along the way, **as Phil King discovers**



So, you've made a cool electronics project for yourself and have posted it online where it has aroused considerable interest and positive feedback? Maybe some people have even said they'd be interested in buying one. At this stage, you might well want

to at least consider turning your idea into kits or assembled products to sell to others. What next?

Developing and bringing a new product to market is a very different affair from creating a DIY project for yourself. As well as needing to modify the design for a manufacturing process – especially if you're intending to go large-scale – and testing prototypes along the way, you'll need to consider a whole range of other aspects, from sourcing components to arranging shipping to buyers.

For a professional-looking product, you will almost certainly need to design a PCB and get it manufactured. You may need to create several prototypes and test them thoroughly. Another key aspect is creating thorough documentation so that buyers will know how to use the product. You'll also need to consider things like testing, packaging, shipping and, of course, pricing – you don't really want to be making a loss!

In this feature, we aim to give you a few pointers on how to get started and avoid some of the main pitfalls that lie in wait for the uninitiated maker. To gain an insight, we've chatted to makers who've made the leap from project to product, along with other experts who are very familiar with the process.

You may not make your first million, but you'll get to create a unique product that other people will enjoy and maybe earn yourself a little cash into the bargain, which could help you fund your future projects. ➤

PARTS & DESIGN

Designing a product for others to use requires careful consideration, and you'll need to source reliable components

Below
Sourcing reliable components has been made more difficult by the global supply shortage

Before you start planning that crowdfunding campaign and large-scale production process just yet, you need to think carefully about your product's design and its potential audience.

"Put some information up about your project on a site like [Hackaday.io](https://hackaday.io) or share it with your relevant groups to get feedback and gauge interest before you invest in buying materials or parts," advises Tindie product manager, Jasmine Brackett. "Input from the community is a great way to gauge interest and build up support before you start selling."

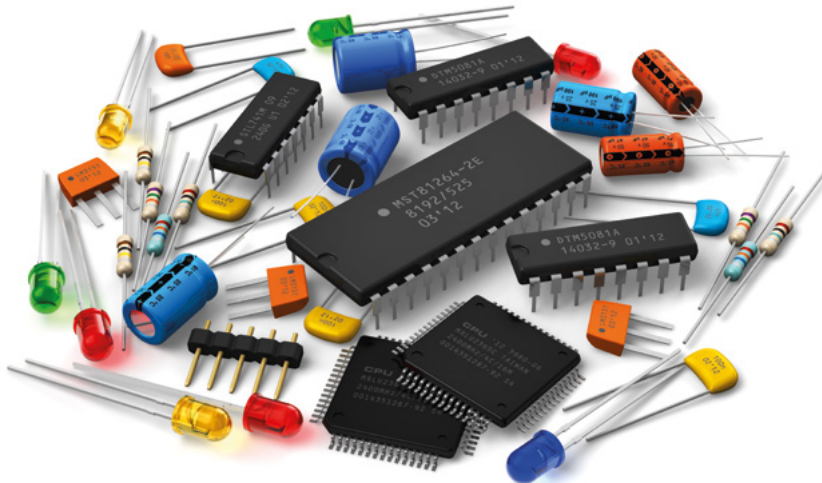
"To make a truly good product, I think it's pretty key to use it 'in anger'," says Pimoroni co-founder Paul Beech. "Get inside the skin of what people will try to do with it. That will tell you a lot about what needs to change, what is most important and needs to be made easier. For the makerverse, I think you have to avoid just making what you think people want. Have a strong opinion about how things should be and make the product reflect that."

"Possibly the biggest [issue], if you're aiming to sell, is knowing exactly who your customers will be. By name. If you want to sell 100 of a kit, make sure you have at least 100 people who will give you a firm yes on being a customer for it. Preferably a lot more. Chucking into the universe rarely works. You need to build a following and interest in what you're doing. You and your approach matter as much as the thing itself."

On a more practical level, you will also need to consider aspects such as whether your product will be a kit and/or ready-assembled, if it will require a custom PCB, and how you will source the components required.

PCB DESIGN

You may well have created your DIY project with whatever bits and pieces you had to hand. Most makers will at least start prototyping using a breadboard or stripboard to connect the electronic components. Naturally, that won't really cut it for a polished product that you want to sell, so – unless you're simply creating a basic electronics kit comprising common components to use with an existing single-board computer or microcontroller – you'll likely need to design a PCB (printed circuit board).



If you haven't taken this step before, we covered the process of creating your first PCB in detail in HackSpace issue 48 (hsmag.cc/issue48), using the EasyEDA software, which is ideal for entry-level electronics. Other PCB design tools available include the more fully featured KiCad and widely used EAGLE (check out SparkFun's tutorials: hsmag.cc/SFEagleTutorials). If you're already familiar with Fritzing, you can even use that to convert a breadboard circuit into a schematic for a PCB.

Whatever tool you use, you will end up with Gerber files for the PCB, solder mask, silkscreen, drilling, and so on, which you will need to send to a manufacturer such as OSH Park or JLCPCB.

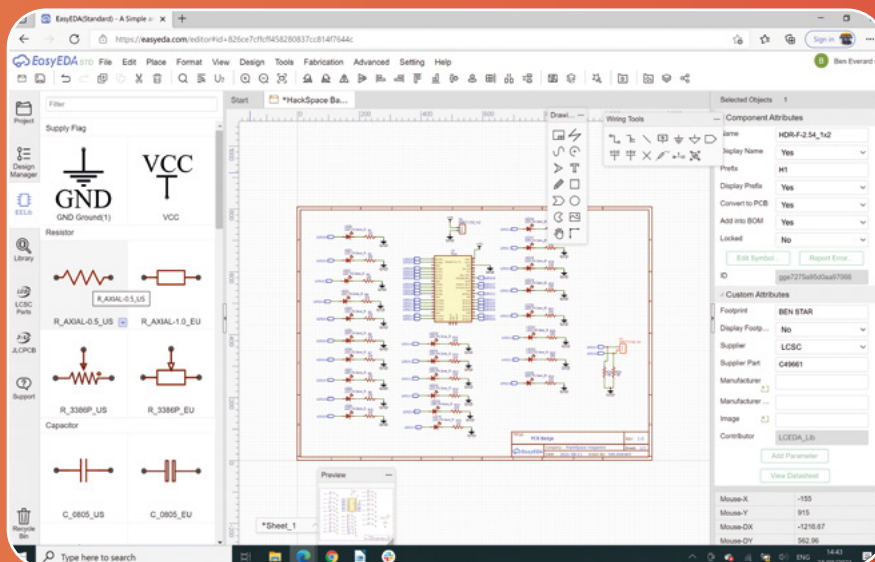
COMPONENTS

Whether you are intending to ship your PCB as a bare board, with separate components in a kit, or with them pre-soldered on the board (by yourself or a manufacturer), you will likely need to source a large number of electronic components.

This is very different from creating a DIY project, as Jasmine notes: "When you're making something for yourself, you can use whatever you have on hand. Unless you're planning to make more than one for yourself, you don't have to think so much about the easy availability or even the reliability of the parts used."

Obtaining readily available, good-quality components can be a challenge at the best

Right
Adding components
to a PCB design in
the EasyEDA tool



BUYING FROM QUALITY VENDORS IS CRUCIAL. IT ADDS COST, BUT IT IS WORTH IT

of times, but has been exacerbated by the current global shortage.

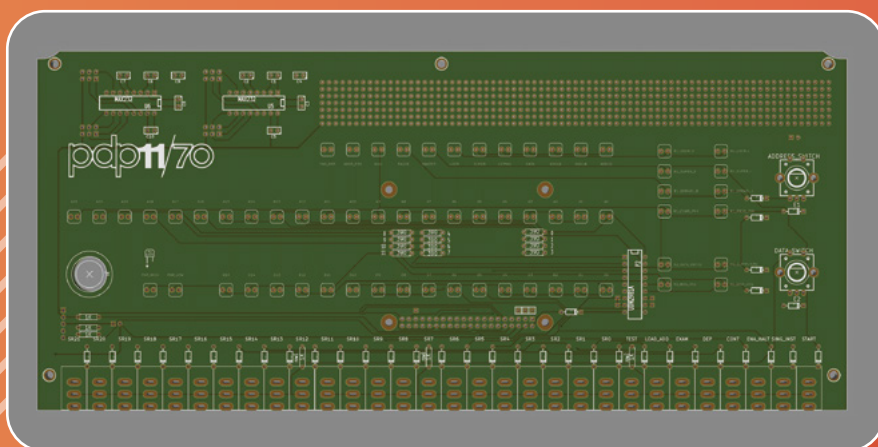
"When you're doing a production run or even a small batch, if you can't get the same parts or their quality varies, you may have to redesign your product to use another component or accommodate variances in sizing or other specifications," adds Jasmine.

And if you're tempted to buy super-cheap components from little-known suppliers,

Oscar Vermeulen (maker of retro computer kits at Obsolescence Guaranteed) offers a word of caution: "I used to buy cheap ICs off the internet – ending up with some chips that did not even have any silicon inside. Buying from quality vendors is crucial. It adds cost, but it is worth it."

In addition, he advises only using mainstream, standard components. "I finished the kit as a product and then found out that this retro-looking switch I chose was really no longer in production. It caused huge problems."

Simplifying the circuit is also advisable, especially if it's for a kit for users to assemble. "If your product is hard to diagnose and difficult to repair, it is probably not something you want to sell as a kit," says Oscar. "This puts a massive constraint on your design options. You better avoid any surface-mount chips, for instance. And every extra component is one more possibility for someone to make a mistake building the kit." →



Left
The PCB design for the PiDP-11/70
replica classic computer kit

THE PRODUCTION PROCESS

Getting a product ready for large-scale production involves prototyping, thorough testing, and quality control

So, you've designed your product. What's next? Jason Coon, of Evil Genius Labs, advises getting prototypes made – by a manufacturer such as OSH Park – before pushing ahead with a final design. “I definitely make mistakes, and sometimes have to fix or alter the design after the first prototype build.

Sometimes it's as simple as missing a label on a PCB. Other times it's a completely un-routed PCB trace that I somehow missed in ERC/DRC.”

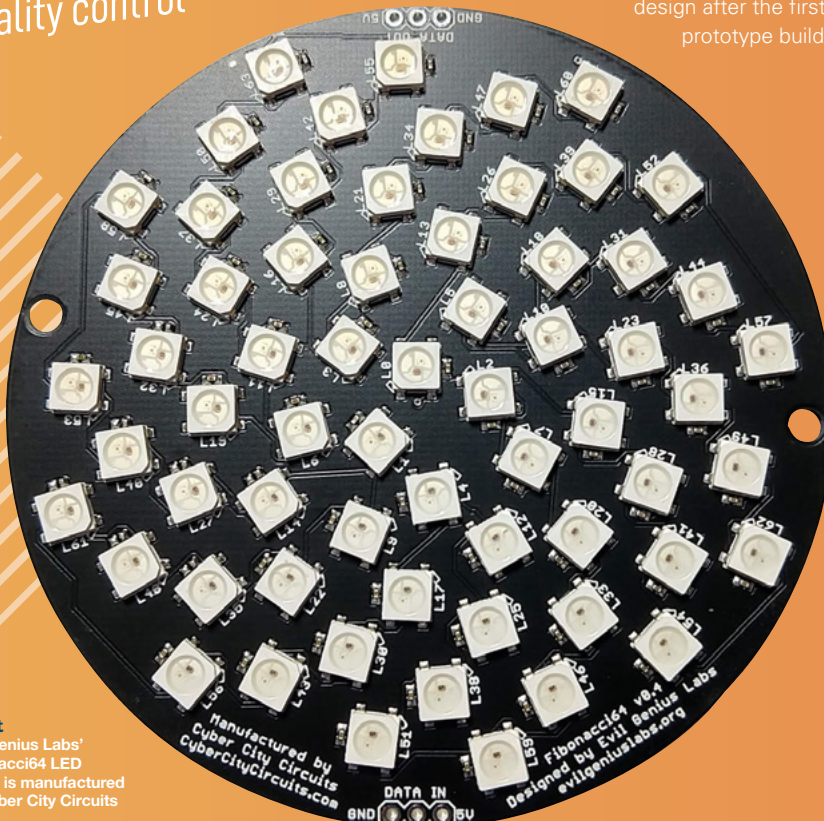
Once you're happy with it, before diving head-first into mass-production, you may want to dip your toe into the water by selling a few units online to ascertain demand, such as in an electronics marketplace like Tindie.

“It's straightforward for people to start slow on Tindie by getting a small batch done, getting feedback, and refining the design or their production and shipping processes,” says Tindie's Jasmine Brackett. “Often, sellers will ‘graduate’ to crowdfunding or mass manufacturing after they have sold on Tindie.” She also notes that it's a “great place” to sell any excess units produced for a campaign.

Unless you want to sink a lot of your own hard-earned cash into fine-tuning the product further and scaling up production, a crowdfunding campaign is an obvious alternative. There are numerous generic crowdfunding services available, such as Kickstarter and Indiegogo, but you may also want to consider Crowd Supply, which specialises in electronics projects and will guide you through the whole process of production and handle a lot of the logistics such as shipping. You'll need your project to be accepted first, however (see ‘Crowd Supply’ box, overleaf).

TESTING, TESTING...

Thorough testing is essential and another key aspect before you sell a product. “I extensively test prototypes of a new design myself, before sending them out to others to help test,” says Jason. “I also test each item individually before packing and shipping it, whether it's an assembled PCB or a fully assembled piece.”



Right
Evil Genius Labs' Fibonacci64 LED board is manufactured by Cyber City Circuits

**Left**

A complete PiDP-11/70 in its case – buyers need to build it themselves as Oscar Vermeulen only sells kits, which are less time-consuming

“Testing up-front, before you start selling is obvious,” adds Oscar Vermeulen. “You do your own testing, have a few friends build the kit, and then you do a test run of a few dozen beta-kits to buyers who know they are the first ones, and they know you will supply them a new kit if something is not right yet. What is less obvious is the quality control of every batch of parts and components that comes in. Cases that are warped, panels that miss mount holes, switches that switch the wrong way. It takes an amazing amount of time.”

GET TO WORK

Making a few units to sell is one thing; large-scale production is quite another. If you’re putting together a kit to sell, it can soon prove very labour-intensive.

“Cutting out or soldering all those pieces might be a labour of love or a way to get what you need cheaper when you’re doing it yourself,” notes Jasmine. “However, when you’ve got to make 10 or 100 by hand and

then test and ship out, you’ll want to find ways to automate or outsource the time-consuming aspects of production, packing, and shipping.

“Most makers get most excited about the design and prototyping phases; the product they love may not seem so wonderful when testing and kitting it. Initially, people often rope in family members or friends to help, but you’ve got to make sure it’s worth everyone’s while; otherwise, it may start to feel like a slog when you’re scaling up.”

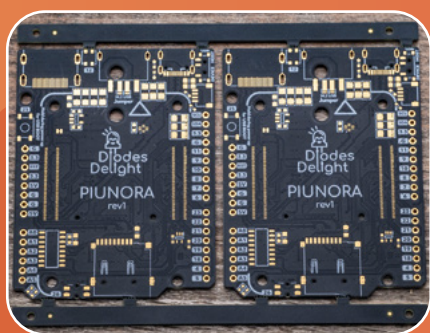
If it all seems too much, the alternative is to employ a manufacturer to assemble your product. “I have some SMT [surface-mount technology] assembly done by contract manufacturers, such as Cyber City Circuits,” says Jason. Oscar is even planning to open a mini-factory to share with other makers developing products (see ‘Maker Factory’ box).

If your product requires a case, laser cutting and 3D printing are feasible options for small-scale production, but are very time-consuming – if you really want to scale up production, you may want to consider vacuum-forming or injection-moulding, as used by Oscar for his replica classic computer kits such as the PiDP-11. “Once an injection mould is made, you cannot correct mistakes,” he says, so you

need to know what you’re doing and get it right first time or it could prove a very expensive error. →

Left

A pre-production prototype of the bare Piunora PCB, straight from the manufacturer

**MAKER FACTORY**

Oscar Vermeulen knows many hobbyist makers are put off by the time-consuming hassle of manufacturing, storing, and shipping stock, so their projects tend to stay at the prototype stage. Which is how the idea arose of opening a mini-factory; along with some friends, he’s planning to open a small-scale facility in Panama, which has great logistics due to the Canal.

“It’s purely for ideas that might be perfectly viable on a small scale – a few hundred or thousand units per year – but just too niche to be picked up anywhere in a commercial manner,” he tells us. “The idea is a co-operative one: share a logistical and administrative infrastructure. Help other makers get their products realised and make their lives easier – avoid filling their garages with parts. Every week, the maker just sends a list of addresses of buyers that need to be invoiced and sent a kit or assembled unit. The rest is done by the mini-factory.”

Staying at a beachside resort, makers will be able to get together with people who know about things like case design, logistics, and production. “The idea is to take a maker product from almost ready to 100% production ready in a one- or two-week sprint on-site.”

Oscar’s mini-factory is set to be built next to his consortium’s small holiday resort, which has a beachside bar – ideal for maker meetings!



DOCUMENTATION & SHIPPING

Once you've got your product made, you'll need to ship it to customers and provide detailed documentation to help them use it

Below

Online documentation for your project is essential, such as this build guide for the PiDP-11/70 replica computer kit

Designing and manufacturing a product is only one part of the equation. You will also need to create comprehensive documentation to show customers how to use it, possibly including a tutorial and accompanying example code.

Of course, you'll need to get your product into the hands of customers first, which entails organising packaging and shipping.

DOCUMENTATION

"An undocumented product is not a usable product," advises Timon Skerutsch, creator of the Piunora dev board. "It is just as important as the product itself and will dictate how well your product will do after

its initial launch. Frustrated customers are not happy customers."

"Documentation is incredibly important, especially for kits, but also for the fully assembled pieces," adds Jason Coon. "It's important to make sure the buyer has a good experience with what they've bought."

It needs to be comprehensive and clear, says Jasmine Brackett. "Especially if there are step-by-step instructions, it's best to get a few people from your target audience to go through it. What seems obvious to you might not be to someone else. It's beneficial to have one place online where all the instructions, files, etc., are kept." This could be a GitHub repo or your own website.

"If you don't include full printed instructions when you ship your product, at least put a link in the box, or even better, on the item," recommends Jasmine. "Then, if you need to update your docs, you don't have to post anything, and folks can find the instructions quickly, weeks or months after they receive the order."

Good documentation includes your product listing. "The clearer your product page is with a good title, description, images, and links to documentation, the better," says Jasmine. "If you don't provide enough information so that the buyer knows it meets the specifications of their project, there's a chance that they may send it back, which will cost you money. It also helps to minimise any pre-sale questions or people deciding to go with another better-documented solution."

PiDP-11: RECREATING THE PDP-11/70

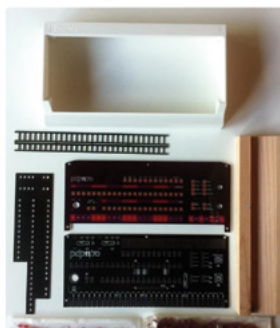
INTRO OVERVIEW TECH DETAIL GET ONE USING THE I1 BUILD THE KIT HACKS FORUM

Building Instructions

Building the PiDP-11 should take about 6 hours. Tools required: Soldering iron, side cutter, Phillips screwdriver.

Parts list

- 1 PCB, 1 case, 1 acrylic front panel, 1 back panel, 1 base mount
- 11 red, 13 magenta **toggle** switches, 1 white, 3 red & 2 magenta **momentary** switches
- 2 brackets (1 item, snap into two) to mount/support switches during soldering
- 64 red 5mm LEDs + 64 LED spacers to mount them higher on the PCB
- 1 shield to push LEDs into their exact space and keep them there
- 37 diodes
- 2 rotary encoders, 2 knobs
- 1 key switch with mounting nut and key
- 1 40-pin extra-height Pi connector
- 1 UDN2981A IC or equivalent & 1 IC socket
- 12 resistors (390 ohm), 6 resistors (1K ohm, or 820 ohm in recent kits)
- 5 M3 screws, 5 M3 nuts, 5 brass M3 10mm hex spacers to mount PCB
- After September 2020: 5 extra M3 nuts to add on the hex spacer, see below





Left
You'll need to think about product packaging, like this example for the Piunora dev board

SHIPPING

Unless you're using a platform such as Crowd Supply that handles all your shipping, you'll need to arrange it yourself.

"It's a big issue that you don't have to think about when making stuff for yourself and that many people overlook," notes Jasmine. "Choose packaging carefully so that it's economical in materials and size to keep shipping costs low, but robust enough to stand up to being tossed around the globe. Consider using a priority flat-rate box if you're in the US and shipping mainly to US customers, as it will save you the cost of buying boxes, plus includes tracking and insurance, which will save you headaches down the road. If you're going to ship internationally, consider self-insuring your packages by adding a little to the product or shipping price. Self-insuring safeguards you

when something unexpected happens to the odd package as you can quickly reship it or refund it without feeling you've made a loss."

In addition, she points out, "Sellers need to make sure they research the local laws and regulations of where they live and look into them in any foreign destinations where they are planning to ship. Depending on where they are based and how much they intend to sell, they may need business licences or to pay tax. Usually, when people start selling, sales come under hobbyist laws, but as they grow, they'll likely reach thresholds to declare their sales for tax purposes." ➔

Below
Make sure your product page has a good title, description, images, and links to documentation

Fibonacci64 Goggle Lens: 50mm disc w/ 64 RGB LEDs

Fibonacci64 Goggle is a beautiful 50mm circular disc with 64 WS2812C-2020 RGB LEDs surface mounted in a Fibonacci distribution.

Designed by Evil Genius Labs LLC in United States of America

Wishlist Tweet Share Pin



\$32.00

Ask a Question

No shipping info available.
Set destination country to see options

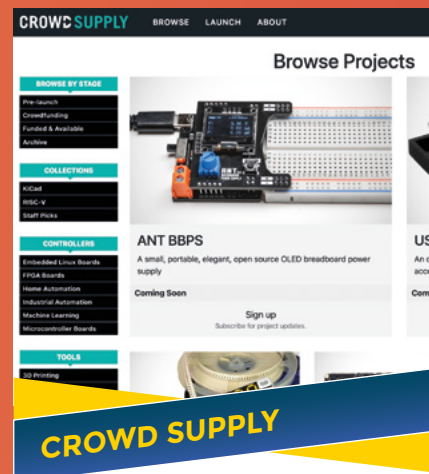
Quantity

1

Diffuser or Lens

The lens is meant for mounting inside goggles that accept a 50mm lens and has no mounting holes. The diffuser is slightly larger to accommodate mounting holes and is meant for use in custom builds, not goggles.

Only 4 left in stock, order now!



Focusing on electronics hardware, Crowd Supply is a crowdfunding service that helps you take your product all the way from prototype through crowdfunding and manufacturing to shipping. So it's ideal for first-time makers, who comprise around 50% of its users.

You'll need your project to be accepted first, however. "It's actually quite strict: we turn away between 90 and 95% of our applications," reveals Crowd Supply's head of community, Helen Leigh. To be considered, you'll need to have a prototype and a commitment to open-source hardware: "We require at least the firmware and schematics to be released."

Once accepted, you're guided step by step through a highly structured process – dealing with batches of issues on a GitHub repo, aided by a product manager. Crowd Supply also provides a photographer to take shots of your final prototype, and a technical writer to help craft the copy for the campaign, which starts with a pre-launch to build up some buzz.

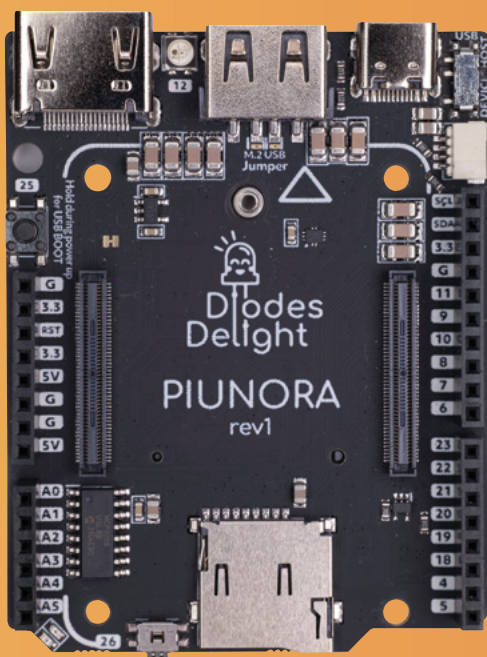
"It is designed for engineers who would not know the first thing about bringing a product to market," says Helen. "And that's why we do all the logistics as well." Aided by Crowd Supply being part of electronics retail giant Mouser, this includes handling all packaging and shipping from a warehouse in Texas. In addition, your campaign's final number of orders from backers will be matched and purchased by Mouser, which will continue to sell this extra stock.

CASE STUDY: PIUNORA

How one maker turned his project idea into a successful product

Below

The final production version of the Piunora dev board, ready for a Compute Module 4 to be mounted on top



When Timon Skerutsch's friend Scott Shawcroft (a lead on CircuitPython at Adafruit) said it would be nice to have the Raspberry Pi

Compute Module 4 in an Arduino form factor, the idea for the Piunora CM4 add-on board was born.

"I started designing it, more jokingly really," says Timon, "but as I went along with the design, I realised that this actually works really well... So I pivoted to actually taking it seriously and thought about what the target audience would be."

Posting the assembled board to Twitter was the spark to turn it into a product: "People really liked it and it got shared a lot. I was always thinking about starting to sell products, but the stress and bureaucracy involved always deterred me." But Piunora felt different: "It was at least for my skill set, an easy product from the R&D perspective. It felt manageable."

MAKING IT HAPPEN

The initial prototype took him "around two days of schematic and board layout, and another day of silkscreen design and detail fiddling. That was about 90% feature

OPEN-SOURCE HARDWARE

Not every maker open-sources their hardware designs, but it's something that should be considered, reckons Tindie's Jasmine Brackett. "As many makers learn from others, open hardware is an excellent resource for learning, and you can create or contribute to projects to get experience. It's a great way for makers to give back to the community."

While some makers may worry about the potential of having their open-source PCB cloned, that's a fairly rare occurrence according to Crowd Supply's Helen Leigh. "Not many people have that happen to them. If you [do], you're very successful already."

Even then, adds Jasmine, "There are many other ways to differentiate your product... You can use good parts, have a high-quality control standard, and provide excellent documentation. Many of the most successful sellers on Tindie cater to a niche, are passionate about that specific community, and are active participants. Because of this, they have a good reputation and can charge for a premium item."

You can find out more about OSHWA open-source hardware certification at certification.oshwa.org



OSHWA is a non-profit organization that promotes the development and use of open-source hardware. It provides a framework for the development of open-source hardware and a platform for the community to share their designs and projects.

WHAT IS
CERTIFICATION?

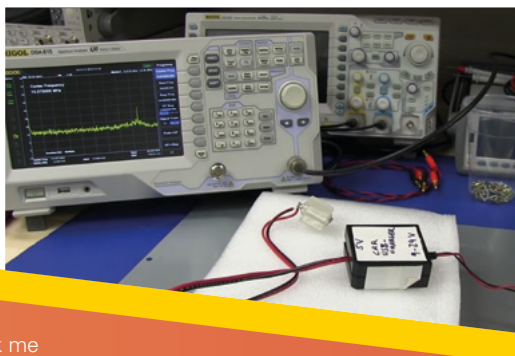
CERTIFY A
PROJECT

SAFETY CERTIFICATION

One question often pondered by makers looking to create products is whether they need to obtain product safety certification, such as CE and UKCA. According to Pimoroni's Paul Beech, "The official line approximates to (for EU/US/UK mainly): if you're putting a product on the market, you'll need to make a declaration about its safety and be able to back it up with documentation. A chunk of that is chasing down materials certification docs from your suppliers. If you're buying from people like DigiKey, they will have the RoHS/REACH docs for the components. So don't fear it, and do the boring legwork."

That doesn't cover emissions/immunity for electronics, however. While most makers won't be able to afford to test their product to CISPR/EN standards in a lab, "the more diligent makers, once they start shipping more than 100 pieces of their small project, do a reasonable chunk of testing with non-lab settings," says Paul. "You can reassure yourself your product isn't going to interfere with Radio 4, which is a good start. Kits aren't a get-out: testing needs to be a genuine 'as the customer will use it' thing. There are plenty of YouTube videos and good articles on 'pre-compliance testing' that are more in reach for small makers."

One of the numerous YouTube videos on pre-compliance testing ([hsmag.cc/EMCPreComp](https://www.youtube.com/watch?v=EMCPreComp))



complete – the famous last 10% took me another four to five weeks over the course of around six months."

That was just for the product design, however. "What took a whole lot more work was setting up production, establishing relationships, doing paperwork, certification, writing emails, logistics to the distributors, creating the campaign, writing software, writing even more emails, documentation, doing PR, making a website, and everything else that goes into making and marketing a product."

From finished prototype to crowdfunding campaign (on Crowd Supply) was around four months, then another three to four months for production to wrap up, and then

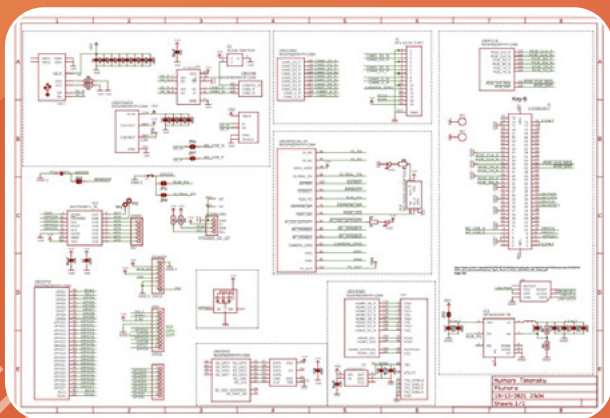
another two months before customers were receiving packages. "All in all, the journey was about a year. If I didn't have my pre-existing experience and relationships [from designing electronics commercially], I think it would have probably taken six months longer."

Timon used Makerfabs to manufacture Piunora. You should see your manufacturer as a partner in your venture, he advises: "They are not faceless robots – talk to them, understand their bias, make clear what you expect and need your product to be... If things don't turn out how you imagined, it's usually a case of not communicating well enough."

WORD OF ADVICE

Timon offers some further advice for potential product makers: "Make sure you've got the bandwidth to commit to it. It will be a lot more

Left
Launching the product on Crowd Supply entailed open-sourcing details of the Piunora hardware, such as this schematic



ADDITIONAL RESOURCES

Need a little more help? There are many online resources providing useful information to aid you in designing an electronics product for manufacturing. Here are just a few...

Hackaday U

hackaday.io/courses

There are a few free courses available on Hackaday, such as 'Prototyping in Mechanical Engineering' and 'Introduction to KiCad and FreeCAD'.

HDDG Talks

hsmag.cc/HDDGTalks

From the Hardware Developers Didactic Galactic meetup group, this YouTube series features some highly relevant talks, most notably 'HDDG 36: Design for Manufacturing'.

Rheingold Heavy

hsmag.cc/RHTutorials

The maker-focused electronics supplier has a range of helpful tutorials, including a series on KiCad.

Contextual Electronics

contextualelectronics.com

A range of hands-on courses that teach you the skills to create your own custom electronics. Membership costs from \$9 a month with a free week's trial.

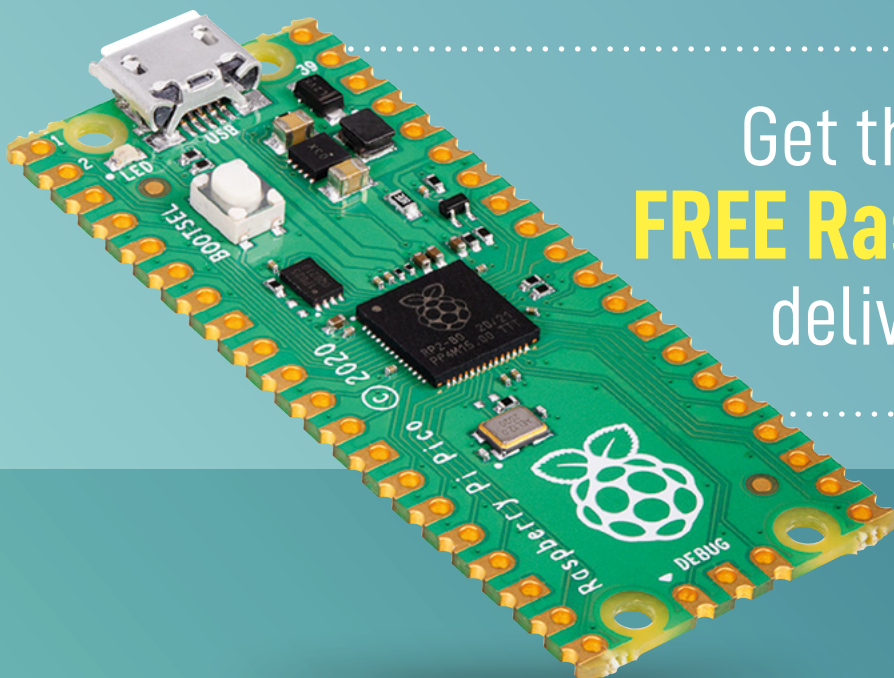
work than you initially imagine, no matter how prepared you are going into it."

In particular, launching Piunora while doing a full-time day job was stressful: "It's possible but it's hard, especially mentally. Make sure you take time for yourself despite the pressure you will be experiencing after you've had your campaign, pre-order etc.... Don't destroy yourself: if you keep being transparent about what's happening then customers will actually be very accepting of delays."

While you don't have to quit your job and do it full-time, "it will definitely alter how you perceive your hobby quite significantly: it won't be just a hobby any more that you do for fun; there will be a lot of things that you need to do that will be very un-fun." ■

SUBSCRIBE TODAY

FOR JUST £10



Get three issues plus a
FREE Raspberry Pi Pico
delivered to your door

..... • hsmag.cc/FreePico

UK offer only. Not in the UK?
Save money and get your
issue delivered straight to your
door at hsmag.cc/subscribe.
See page 66 for details.

Subscription will continue quarterly unless cancelled

SUBSCRIBE on app stores

From £2.29



 Buy now: hsmag.cc/subscribe

Free Pico with print subscription only



MAKE | BUILD | HACK | CREATE

HackSpace

TECHNOLOGY IN YOUR HANDS hsmag.cc July 2022 Issue #56



**SAVE
44%**



3D PRINTING POLYHEDRA SUBLIMATION FIREFLIES



HOW

By Rob Miles

I

MADE

MAKE A 3D- PRINTED GAME WITH PYTHON

Write a Python program to design play pieces, and then create compelling gameplay with them

This article describes how to make 3D objects from Python code and then design game mechanics that deliver compelling gameplay. A

completed game, 'Tag Jousts', shows how this is done. You can find all the designs and sample programs, plus printable copies of the rules, here: tagjousts.xyz. You can use this to make your own copy of the game, or use the designs as a starting point for your own game development.

THE TAG JOUSTS GAME

Figure 1 shows a set for two players. The game contains elements of skill, bluff, and luck which makes it quite fun to play. You could play the game with tokens made of card or paper, but we are going to discover how to use software to create the designs of the play pieces. We are going to use the Python programming language running inside FreeCAD.

FIND YOUR INNER PYTHON

Large programs are built as several co-operating components. A word processor will have a text editor, a spell checker, and a printer driver. These parts are usually 'baked together' into the finished program. However, some applications are constructed as frameworks which support elements loaded as the program runs. FreeCAD works like this. It contains lots of Python routines which

What I used

- ≥ A computer that can run the FreeCAD program. FreeCAD is a powerful computer-aided design (CAD) package. It is available for PC, Mac, and Linux-based systems. You can find it here: hsmag.cc/FreeCAD
- ≥ You can print your game pieces using any 3D printer. If you just want to create gameplay, you can also use paper and card game elements, or even blank playing cards to construct your games.

provide all the various behaviours. Actions you perform in FreeCAD (for example, clicking the Cube icon in the editor to add a cube to a design) are implemented as actions on Python objects inside the program. You can see this in action by opening the View menu and selecting the Panels item. Then set the Report View and Python Console items to have ticks against them. Now, whenever you perform an action in the program, you will see the Python components at work.

Figure 2, overleaf, shows the code that runs when you create a cube and a cylinder in FreeCAD and then fuse them together. These actions were performed by clicking the Cube and Cylinder icons in the Part editor, then selecting both items in the drawing and selecting the 'Make a union of several shapes' icon. These statements may look complicated, but don't worry. The ones we will use in our programs are a lot simpler.

Using Python statements to control an application makes it very easy to create macros. A macro is a sequence of commands you might want to repeat. We could create a 'Make me a cube and a cylinder' macro by selecting the Recording option from the Macro menu and specifying a file to hold the macro. We then add a cube and a cylinder to the drawing as before, fuse them together, and select 'Stop macro recording'. When you

create a macro, it is saved in a text file with the language extension .FCMacro.

Figure 3, overleaf, shows the Python statements in a macro that creates a cube fused to a cylinder. They look remarkably similar to the ones in **Figure 2**. Macros can be loaded and edited within the FreeCAD environment, but the code editing support is very limited. Debugging is impossible, and there isn't even any way of searching the source code. →



Figure 1 ♦
The tag designs are easy to print on a printer such as an Ender or Prusa. A set takes around three hours to print at good quality

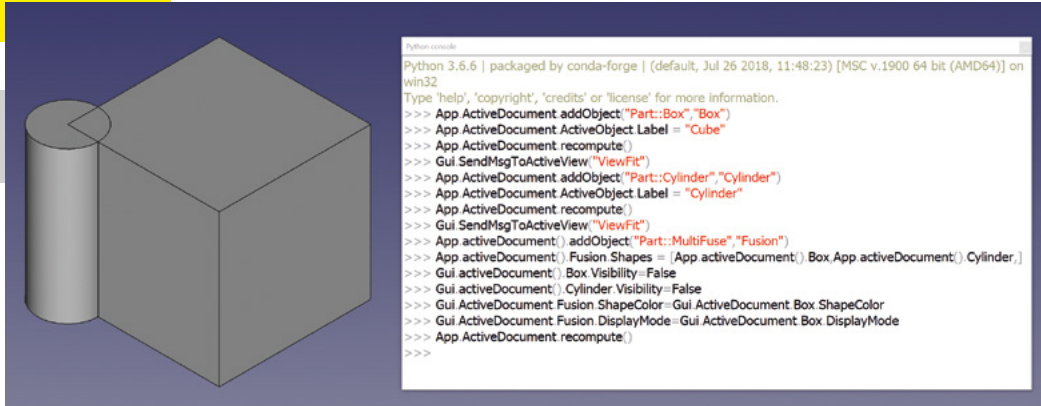


Figure 2 You could create your FreeCAD objects by typing everything into the Python console, but it is much easier to generate the commands using the graphical user interface (GUI)

FAKING FREECAD

If you want to write your macros more easily, there is a framework you can use to do this. It takes the place of the FreeCAD macro environment and lets you create your code directly inside Visual Studio Code. It provides a subset of FreeCAD commands, but it does make it much easier to write and debug code. You can find the framework at hsmag.cc/FreeCADSimulator.



We could make our game objects using the high-level FreeCAD commands we have already seen, but it is much easier to use lower-level ones. **Figure 4** shows how these are used to create a box and a cylinder and then fuse them together. The **tag** objects we are going to create are made up of stacked cylinders and boxes, but you can make much more complex items this way.

When we create an object in FreeCAD, it has a particular size (width, depth, and height), position (where it is in the design), and orientation (which way it is pointing). These values are expressed as X, Y, and Z values in millimetres.

The objects in **Figure 4** were created with the dimensions shown in the code and default position and orientation. The bottom left-hand corner of the box and the centre of the cylinder were placed at position $x=0, y=0, z=0$ and oriented on the Z axis. You can see this because the cylinder is centred around

the bottom left-hand corner of the cube. Objects expose methods that can be used to translate and rotate them.

Figure 5, overleaf, shows how a program can position objects in 3D space. Note that if we attempt to put an object 'inside' another one, this will not cause a problem. We can make holes in objects by moving one object inside the other and combining the objects using the Cut operation.

Figure 6, overleaf, shows how we can make a 'bead'. Note that although the hole is the same size as the pin on the peg that we created in **Figure 5**, the bead might not fit onto the peg. You would have to make the hole larger or the pin smaller (or both) to be sure that they would fit together.

MAKING TAGS

The code in **Figure 7**, overleaf, will generate one tag. The great thing about making tags in software is that we can use looping constructions to create lots of items if we want to.

Figure 8, overleaf, shows how nested **for** loops can be used to create many objects. The pattern is created by combining lots of rotated cubes. Each cube is rotated slightly to make the pattern more interesting. The


```

1 # -*- coding: utf-8 -*-
2
3 # Macro Begin: C:\Users\rsmil\AppData\Roaming\FreeCAD\Macro\cylinder and cube.FCMacro ++++++
4 import FreeCAD
5 import Part
6
7 App.ActiveDocument.addObject("Part::Box","Box")
8 App.ActiveDocument.ActiveObject.Label = "Cube"
9 App.ActiveDocument.recompute()
10 #Gui.SendMsgToActiveView("ViewFit")
11 App.ActiveDocument.addObject("Part::Cylinder","Cylinder")
12 App.ActiveDocument.ActiveObject.Label = "Cylinder"
13 App.ActiveDocument.recompute()
14 #Gui.SendMsgToActiveView("ViewFit")
15 App.ActiveDocument().addObject("Part::MultiFuse","Fusion")
16 App.ActiveDocument().Fusion.Shapes = [App.ActiveDocument().Box,App.ActiveDocument().Cylinder,]
17 #Gui.activeDocument().Box.Visibility=False
18 #Gui.activeDocument().Cylinder.Visibility=False
19 #Gui.ActiveDocument.Fusion.ShapeColor=Gui.ActiveDocument.Box.ShapeColor
20 #Gui.ActiveDocument.Fusion.DisplayMode=Gui.ActiveDocument.Box.DisplayMode
21 App.ActiveDocument.recompute()
22 # Macro End: C:\Users\rsmil\AppData\Roaming\FreeCAD\Macro\cylinder and cube.FCMacro ++++++

```

Figure 3

You can uncomment the GUI statements shown in green if you want the actions of the macro to be displayed on the user interface when it runs

die displays a blank face, the attacker can decide what will be scoring and says either 'squares' or 'circles'. All the players then play a tag in the joust. Once the tags have been chosen, the players reveal the underside of their tags and score them by counting the total number of the chosen shape on the top and bottom of their tags. The player with the highest score wins the joust. The winning player captures their opponents' tags and discards their attacking tag. If the top score in a joust is tied, all the tags in the joust are discarded. Play continues until each player has run out of tags. The winner is the player who captures the most tags in jousts.

```

1 import Part
2
3 box = Part.makeBox(10,10,10)
4 cylinder = Part.makeCylinder(2,10)
5 result = box.fuse(cylinder)
6 Part.show(result)
7 Gui.SendMsgToActiveView("ViewFit")
8 Gui.activeDocument().activeView().viewAxometric()

```

import the Part module
make a box 10 wide, 10 deep and 10 high
make a cylinder radius 2 and height 10
fuse them together
show the result
scale the display
set the view

Figure 4

The Part module contains functions that can be used to make boxes, cylinders, spheres, and other shapes

rotate function is given the point to rotate about (the axis of rotation) and the number of degrees to rotate. You can make the pattern even more interesting by rotating each cube by a different angle or making the boxes different sizes.

THE TAG JOUSTS GAME

We can start our exploration of game-making by looking at the Tag Jousts game, which uses 3D-printed components and has a simple but engaging gameplay mechanic. Each player in the game has a set of 16 tags and a special die.

Figure 9, overleaf, shows one of the game tags. Every tag has a different combination of two shapes (square or circle) on the top and two on the bottom. The counts of the shapes on the tag determine the 'scores' of that tag. The game is played over 16 rounds. Players take it in turns to challenge each other in a 'tag joust'. At the start of a joust, the leading player throws their die to decide what will be scoring for that turn. If the die shows a square or circle, that shape will be the scored one for the round. If the

THE GAME DEVELOPMENT PROCESS

The development process for Tag Jousts turned out to have a lot in common with video game development. This shouldn't be surprising, in that the aim of both video and board games is to provide an experience in which the players have a set of inputs that they can control and an aim to achieve, whether that is 'shoot all the aliens' or 'acquire the most tags'. The game designer needs to decide what the player can control, how the players interact, and what random factors to introduce to make the gameplay more interesting. This all goes to make up what we can call the 'gameplay mechanic'. At the heart of all this is playtesting – discovering what works and what doesn't. As with video games, it is very important to make something for people to play with as quickly

as possible to discover what works and what doesn't. Let's look at the development process for Tag Jousts.

ACHIEVE BALANCE

In the first version of the game, the number of squares and circles on the top and bottom

of each tag was set randomly, and each player had a set of random tags. This worked quite well, but there was concern that some sets of tags would be more powerful than others because they had more tags with →

“EACH PLAYER IN THE GAME HAS A SET OF 16 TAGS”

Figure 5

A vector contains the values of X, Y, and Z, respectively. The cylinder is moved 5 in the X direction, 5 in the Y direction, and 10 up so that it now sits on the top of the cube

```
C:/Users/rsml/sources/repos/GitHub/BattleTags-game/3d printing/peg.FCMacro - Editor
1 import Part # import the Part module
2 from FreeCAD import Base # import the Base module
3
4 box = Part.makeBox(10,10,10) # make the box
5 cylinder = Part.makeCylinder(2,10) # make the cylinder
6 cylinder_move = Base.Vector(5,5,10) # vector containing move direction
7 cylinder.translate(cylinder_move) # move the cylinder
8 result = box.fuse(cylinder) # fuse them together
9 Part.show(result) # show the result
10 Gui.SendMsgToActiveView("ViewFit") # scale the display
11 Gui.activeDocument().activeView().viewAxometric() # set the view
```

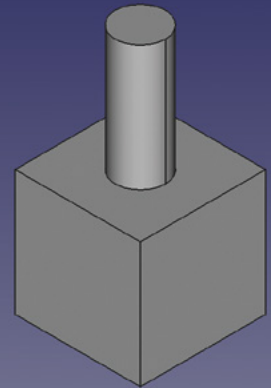
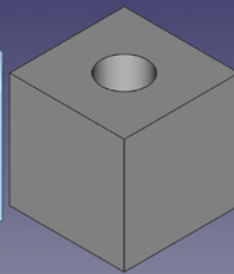


Figure 6

```
C:/Users/rsml/sources/repos/GitHub/BattleTags-game/3d printing/peg.FCMacro - Editor
1 import Part # import the Part module from FreeCAD
2 from FreeCAD import Base # import the Base module from FreeCAD
3
4 box = Part.makeBox(10,10,10) # make the box
5 cylinder = Part.makeCylinder(2,10,Base.Vector(5,5,0)) # make the cylinder at (5,5,0)
6 result = box.cut(cylinder) # cut the hole
7 Part.show(result) # show the result
8 Gui.SendMsgToActiveView("ViewFit") # scale the display
9 Gui.activeDocument().activeView().viewAxometric() # set the view
```



The balance that we have now, with a player getting to choose the scoring shape around a third of the time, seems to work well. With more playtesting, it might turn out that this needs to be adjusted, in which case we can change the number of blank faces on the die.

CO-ORDINATE CONFUSION

CAD tools and computer software use X, Y, and Z values to express 3D positions, but they do it in different ways. When you're designing with FreeCAD, you will find that the Z axis comes up out of the floor. You can think of X as the width of an object, Y as the depth (moving away from you), and Z as the height. However, if you design a graphical user interface in Python using the Tkinter GUI library, you will discover that X is still the width of the object, but Y is now the height up your screen, and Z is the distance 'into' the screen.

Figure 6

We can add a vector to the parameters for the 'makeCylinder' function to position the cylinder when it is created. The Vector class is defined in the Base module

higher numbers of squares and circles, so it was decided that every player should start with the same set of tags. The number of squares and circles on the top and bottom of the tags was reduced to two so that each player could have a set of 16 tags made up of every possible combination of squares and circles on top and bottom.

ADD RANDOMNESS

Playing with the same set of tags made the game completely fair, but it also turned out to be a rather flat experience. It was felt that a little randomness would make the game more exciting. Some games (snakes and ladders) are entirely driven by chance, whereas others (chess) are entirely strategic. What we wanted was a game that had a balance of chance and skill. The selection die was added to help with this. This makes it possible for a player to be strategic, but for an underdog to win if they are lucky enough (i.e. if the selection die comes up in a shape that favours their tags, or lets them choose).

SET A SCENE

Once you've had an idea of how a game can be played, it is interesting to consider different scenarios for the gameplay. Tag Jousts is quite abstract in nature, as is the idea of scoring using squares and circles. Players seem to quite like this, and they also like the designs of the tags and how they are used in play. If you look at games on sale (and there are many), they will have all kinds of different scenarios which are wrapped around a gameplay mechanic. We could use 'red lasers' and 'blue lasers' rather than circles and crosses, and we could replace the tags with model robots holding different coloured laser guns and put coloured ammo packs on the bottom of

**"RANDOMNESS
WOULD MAKE THE
GAME EXCITING"**

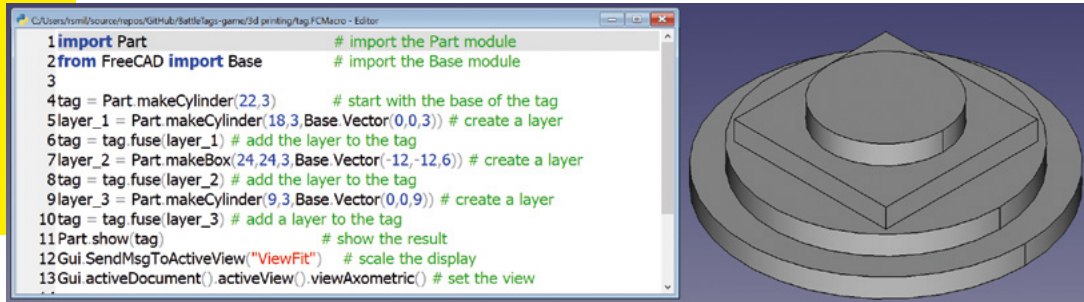
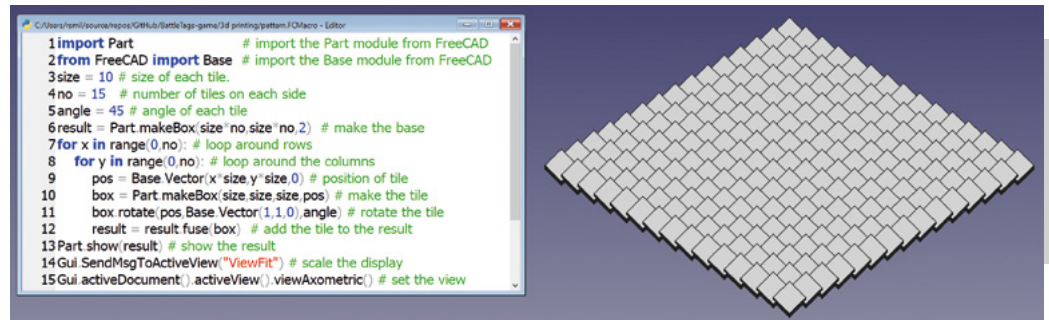


Figure 7 ♦
The number of circles and squares on the top tag are part of the 'score' for that tag. This tag would be worth at least two circles or one square

Figure 8 ♦

You can make lots of different patterns by changing the values in this program. Note that a large pattern like this takes a long time to render; you can try out ideas more quickly by reducing the size of the pattern



each model. Then we could call each round a duel and change the name of the game to 'Super Laser Robot Shootout'. It is quite fun to take a game and consider different scenarios in which it can be played.

MAKE A PRODUCT

Your friends will be much more inclined to play your games if you make them as close to a 'real' product as you can. This takes surprisingly little effort. First, you can consider packaging. If your game uses small tokens, you can put them in a bag. Small bags can be bought cheaply from well-known online retailers. Search for 'hessian party bag'. Next, you can create some printed rules. You can take the STL files containing the 3D designs from FreeCAD and drop them straight into word processors and graphics packages and put some text around them. Be sure to pay attention to design – choose a nice font and use it consistently. Print the rules on coloured paper; this makes them stand out nicely. Finally, you can create a website for your product. This author managed to pick up the domain **tagjousts.xyz** for a year for less than a pound from **namecheap.com** – you can use GitHub to host the site for free. Pop the rules and any other information on the site, and you now have product support in place.

KEEP DEVELOPING

Once you have a game that works, you can think about changes to make the game even more interesting or expand the configuration. The initial version of Tag Jousts was envisioned as a two-player game, but it turns out that many more can take part and the game still works. We might decide it would be interesting to allow a player to select a 'squad' of tags for a particular game. We might add some 'joker' tags that can be circle or square. We might add some new shapes. And then we might discover that inventing a game while playing it is at least as much fun as playing a game that you bought from a store. □

TAGS OF FUN

The game grew out of key chain accessories printed and sold in aid of Red Nose Day, a UK charity event. Each 'tag of fun' has a random (and probably unique) distribution of squares and circles on the top and bottom. You can find the code to create tags of fun here: tagjousts.xyz.

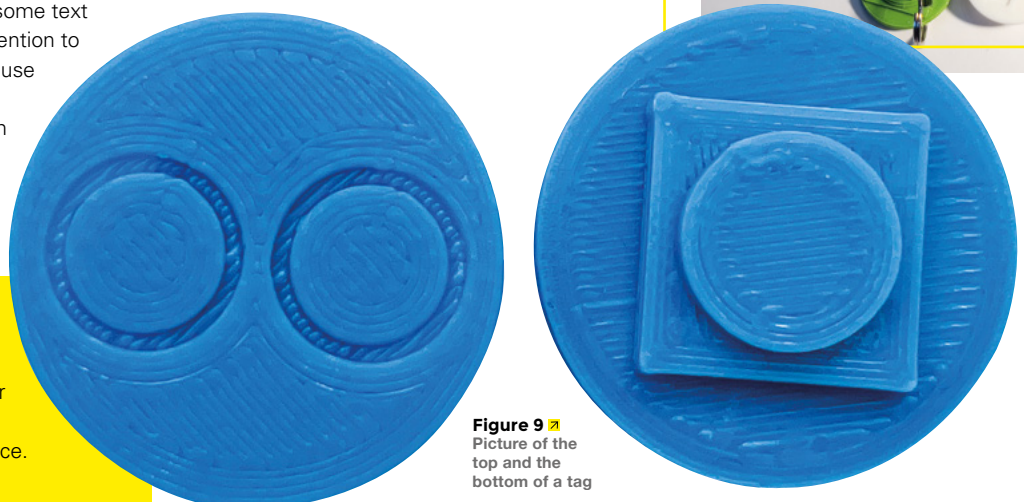


Figure 9 ▣
Picture of the top and the bottom of a tag

HackSpace magazine meets...

Alex Glow

Hackster's Lead Hardware Nerd shows how not to create a load of e-waste

A

s the Lead Hardware Nerd at Hackster.io, Alex Glow has an insider's view of the technology that's coming down the rails at us in the near future.

Happily, she also makes it her business to share that knowledge, supporting a community of almost two million hardware tinkerers, inventors, and improvers. She's also the brains behind a good few projects herself: she's sent music into space, was an early adopter of the companion robot, makes projects with LEDs and PCBs, and... well, whatever she feels like really.

Which made Alex think: how do I keep doing all this, without generating a load of e-waste, burning a ton of carbon, and sending a load of 3D plastic into landfills? The result of this line of questioning is **green-ee.com**, a new (but already extensive) resource for the discerning maker who doesn't want to turn Earth into a fiery rubbish bin. Read on for more! ➔

Right ➔
Alex Glow, Lead Hardware Nerd at Hackster, has built a community of over 2 million makers

Image: Alex Glow
(alexglow.com)



Hackspace You revel in the glorious job title of Lead Hardware Nerd. What does that entail?

Alex I've made up that job title, obviously. I was the first employee at Hackster, which is a community of hardware developers that has been growing really fast and is now very large – we've got almost two million members. I started out as the community manager, someone who was drawing people to the site by reaching out to people finding cool projects and inviting people to join. And I was also publishing my own projects on there.

At the time, I was working for a company called Pinocchio as their maker in residence. And they're basically a mesh network for Arduino that has since pivoted and become Filament, which is like an industrial version. I was publishing projects for Pinocchio and working at Hackster at the same time. After a couple of years, our co-founder was like, 'We should do videos!' And I was like, 'Yes! I'll do one a month!' And he was like, 'How about one a week?' I'd never done video before, so the early ones are extremely amateurish – I like to think that I've gotten better over the years. I think that cringing at your earlier work is a great way to see that you've grown.

There's also this thing that Ira Glass calls the taste gap – where you want to create things, and you have a certain sense of taste, but you also know that what you're making right now is kind of crappy.

Anyway, what I do day to day is I make videos introducing people to new technologies that they can use to make electronics, interview people who are doing awesome stuff with electronics, and build projects and write tutorials. I also give talks and run events sometimes. [Recently], for example, I was at Embedded Vision Summit, just running around with a camera.

HS You must see loads of interesting projects on a day-to-day basis. What are the cool kids working on nowadays?

AG The cool kids are working on Green EE, as I started calling it because I haven't got a better name: building technology in a sustainable way. Technology that is helpful and thoughtful.

There's so much interesting stuff, but the more frivolous answer is that I think there's a lot of cool stuff going on with keyboards, and a lot of cool stuff with PCB art. There's also a growing interest in companion bots – of course, I'm paying attention to that.

But yeah, I really am excited about the stuff that people are doing with more sustainable technology. And there are two aspects to that: there's building things in a sustainable way, and then there's thinking about what to build – for anything to be truly sustainable, it's got to be helpful and not just built as a learning exercise. There's plenty of value in building things to learn –

“

I think that cringing at your earlier work is a great way to see that you've grown

”

that's how we get people excited to build things that are more helpful. Once you get excited about using your skills and passion to help the world, then you can think about building things specifically for good purposes. My work at Hackster is itself part of this. I love it because it's a worldwide community sharing open-source designs for electronics, spreading hardware knowledge, and I frequently get to promote projects that combat poaching, detect forest fires, and help in other ways!

HS So how do you make your projects more sustainable? We're running out of resources you know.

AG There are some really obvious ways around this. iFixit (which is an amazing resource), has this great line: 'the most

sustainable phone is the one already in your pocket'. So really, thinking about whether you build the thing in the first place, I think, is the first step.

Once you get into the nuts and bolts of it, things like using lead-free solder, researching what materials work for your use case, and using recycled or biodegradable materials. There's some really amazing work going on with PCB substrates. Neither of these is ready yet for public consumption, but Jiva Materials is doing a cellulose-based PCB substrate, so it's not only recyclable and biodegradable, but it's easier to peel-off the components and the traces and things. Instead of having to peel that stuff off manually or by machine, you can just degrade out the central substrate and it all just falls apart, which I think is really cool.

There's also one that's based on chitin from shrimp shells. Shrimp shells are a waste product; they take these and grind them up, and they turn the chitin in them into PCB substrate. This project was at a very early stage when I saw it, and I don't know if they have actually done it yet. But ideas like that, I think, are really cool.

As for practical advice for the here and now, using cardboard to prototype – it's cheap, recyclable, and it's a waste product. And there are also considerations of how you power the thing you've built. I've gotten fascinated by this joule thief circuit. It allows you to power low-power projects from a supposedly dead battery. It uses a feedback loop building up a magnetic field in a ferrite toroid, with the low power that you're drawing from the battery over time, and decomposes rapidly and dumps all that magnetic energy back into the circuit as electrical energy and adds that to the power from the battery, and therefore is able to power your little LED or whatever for a short burst – then it goes back into the building-up phase.

Obviously, there's a lot of cool LED stuff that we can do with a circuit like that – bike lights, for example. That's an added bonus – a project that helps people stay →



Left  Cardboard is the ideal material for prototyping – it's light weight, easily workable, and cheap

Image: Alex Glow
(alexglow.com)

Alex Glow

—
INTERVIEW

Below ↗
Alex's creation
F3NR1R is a
shoulder mounted
companion bot – a
robotic familiar

Image: Alex Glow
(alexglow.com)



safe at night on a sustainable method of transportation. But then, you can't use that to power things like microcontrollers, unless you have some kind of smoothing.

People have also built circuits for this that allow you to charge, for example, a five-volt battery off of it, and then you can use that to power your project. So I think that power systems are very interesting. There's a laptop called the MNT Reform that uses a different type of battery – it uses lithium iron phosphate chemistry, which is less harmful [than standard battery tech]. It doesn't use cobalt, which is a conflict mineral. It doesn't have as much capacity, but it has more cycles, so it's better in some ways – it's a trade-off.

HS Then I suppose you can also make hardware more sustainable by making it easier to repair when it breaks.

AG Exactly. iFixit has guidelines for designing your product so that it can be easily repaired. So in that way, you're sort of making it more sustainable.

I gave a talk a few years ago about what I called open-ish hardware, which is sort of the same idea as using screws instead of adhesives, and doing what some companies do, which is abrade the ID markings off components so that users can't replace them. That's obviously an anti-repair, hostile thing – we can sort of make choices that are better.

There's a lot of really cool stuff going on with updating older technology, like Joey Castillo's sensor watch. So, if you have an old Casio F-91W, you can put a new world in it.

Transparent electronics are having a comeback right now. You know, the whole 1990s revival. And I think that's really cool as well. It obviously helps people who

want to learn about electronics, but it also helps those who want to repair stuff, because you can see inside.

HS I saw a talk you did in which you said that the degradation of the environment wasn't our fault, but it is our responsibility. What do you mean by that?

AG I love the idea of building electronics, and I want to spend my life doing that. Though, there are lots of other things I love, too. A lot of the things that we have are part of our human lifestyles nowadays – like [mobile] phones, for example, [they're made] in a way that's really harmful. We also know that a lot of our clothes are produced in damaging ways. However, we do have choices that we can

Transparent electronics are having a comeback right now. You know, the whole 1990s revival

make, depending on our budget, our level of privilege, and our access to those choices. For example, it's easier for someone with the means to shop organic or support local businesses, [because they'll] pay a premium for that.

Newer technologies that are designed with an eye on sustainability and fair labour conditions are going to cost more, just because they're new, or they're harder to source, or like, you have to pay people a living wage. Anyway, these choices are often hard to make and hard to adopt. And they require extra research because they're not as widely distributed. And you know, there aren't big corporations putting out free tutorials on this stuff, because that's not what benefits them. And so we live in this ecosystem where it's hard to make good

choices, and those choices are sometimes just made for us. But often, we're told that the solution is personal action. And that can be really demoralising for a couple of reasons, obviously, because on one level, it's hard to see your individual impacts when you can't see an immediate change from that. It starts to feel like, what's the point? It's not actually changing anything. But that's designed into the system as well. As we've seen with BP [which invented the concept of the personal carbon footprint], there's a reason that an individual person taking one fewer car journey per year doesn't necessarily show up at all.

A lot of this stuff is focused on choices that huge corporations are making, and our leaders or governmental leaders are making when they could make better choices, but the focus has been diverted onto us as individuals. And so, we're being made the patsies in their desire to not change. If we can release ourselves from that guilt, that will make change more possible.

Take, for example, when you're being shamed, or being made to feel guilty: for some people that's motivating, and for others, it's just not. We have enough of a problem with mental health right now, I don't think we need to add to that when we don't need to, and it's not helpful, and it's not true. We have power to change the conversation, and to make choices, and spread that information and use our own experiments, share what we've learned, and make that accessible to everyone.

The fact that we have that power is kind of amazing. And then the more that we share with people who are coming up, like young people, students, people who are going to be building stuff tomorrow, as well as today – those are the people who are going to be in the companies tomorrow, and be able to have this knowledge and provide a starting point to make better choices for the companies.

Obviously, we're all still beholden to whoever's above us. For example, at Amazon, workers have advocated →



Below
Even when the electronics fail, these PCB earrings will look damn good

Image: Alex Glow
(alexglow.com)

HS So is that what you're hoping to achieve with Green-EE.com? More power, more choice, more information?

AG Absolutely. I've been asking myself the question, 'How could I, in good conscience, start something where I'm selling electronic things and not feel like I'm being a jerk?' And my limbic brain comes back with the answer that the best idea is to just not build it, because you're contributing to the problem by building stuff. But I want to try and look deeper and turn this into something that we're all learning together. I was amazed that a resource like this doesn't already exist, honestly. It came from this talk for the Open Hardware Summit. The first couple of months that I was working on the talk, I was asking people in climate action, tech, and other communities, 'if this is repeating someone else's work, let me know. And I'll just, like, link to that'. And there was no response.

I still haven't come across anything that's like this project, but I still think it must be out there. If there are more [projects] like this, I would love to hear about them. iFixit has a lot of incredible resources, but that is specifically focused on repair, which is just one very important aspect of this. Then there's

time and time again for better practices. And time and time again, the shareholders have shut them down. There's only so much that you can do. But we can build alternatives so that people are able to create different options for others to choose. And then there's people who can boost the signal... everyone has a certain thing that they can do, you know – some people are developers; they can build these options. Some people work with hardware; they can build those options. Some people have access to funding, while others have access to publicity and other stuff. And in that way, we can give people more choice, make them feel more empowered and stop guilt-ing ourselves, but also realise that we have this power. It's beautiful.



things like auto-certification, and things like open-source designs for assembly and stuff, and information on the business side. And there's a huge amount of content from people developing websites on like, green or web dev. And that's really interesting. And I've sort of put some of that on the site as well. The goal is to just provide a starting point for people and link out to a lot of the stuff that already exists out there – it's just a matter of making it more accessible. It's me in the present day building something that I will find useful in the future, and that I would have found useful a few years ago.

HS Are you optimistic that people are awake to doing the right thing?

AG There's that William Gibson quote: 'The future is already here – it's just not very evenly distributed'. We have horribly wasteful things, like dumping potatoes into the sea to preserve the market demands. But there's a lot of exciting things going on as well; things people are doing in collaborating with other species. There's a lot of work happening with fungi, and how you can use them for bioremediation when there are oil spills or heavy metals. And you can use them for building materials, like self-healing construction materials. I wish that I liked eating mushrooms, because I just hate them. But I love them in every other way – I think they're fascinating.

There's a building in Hamburg that is partially powered by these algae panels on its exterior, which is incredible. People are using algae for producing oxygen and eating carbon dioxide – which is what the ocean does, right? I think there's a lot of promise in the idea of biomimicry, where we're looking at the solutions that the Earth already uses for these things and emulating those. I'm not an expert – I'm just trying to find these things out and share as I go. [□](#)

Right [↗](#)
Alex's art – inspired by Yuri Gagarin among others – has been in space on satellite 0c66

Image: Planet
(planet.com)





CONCRETE

If you want to make something that lasts, build it from sand and cement. Rosie Hattersley suggests making concrete plans



Rosie Hattersley

@RosieHattersley

Rosie Hattersley writes tech, craft, and life hacks and tweets @RosieHattersley.



Given its use in everyday building projects, concrete seems like a fairly modern material, but it's actually been around for thousands of years. The Romans

came up with the formula involving cement, water, sand, and gravel that we recognise as concrete, using it for everything from aqueducts and temples to the Pantheon. At a mere 1900 years old, this beautiful building is a concrete new-build compared to the Egyptian and Turkish subterranean water stores and structures that date back 8000–12,000 years. Robert Courland's *Concrete Planet* is a comprehensive guide to this durable material and provides plenty of inspiration for creative uses:

hsmag.cc/ConcretePlanet.

"THE ROMANS CAME UP WITH THE FORMULA INVOLVING CEMENT, WATER, SAND, AND GRAVEL THAT WE RECOGNISE AS CONCRETE"

Concrete lends itself incredibly well to use as a sculptural material, whether that's large undulating expanses for wide arches, or a skatepark, Brutalist skyscrapers, or large cast forms, as per the designs

by modernist artists such as Barbara Hepworth or Henry Moore. Its fine particles mean it can pick up fine detail and texture, which artist and maker Made By Barb exploits in her botanical designs, on the facing page.

Concrete's widespread use in construction and DIY projects means there's every chance you may have a leftover bag of cement from a home improvement task that could potentially become a concrete-based project. As with many building materials, a supply shortage of the cement used to make it has pushed prices up, which is one more argument in favour of trying to construct some concrete items yourself rather than pay even more for a builder to do so.

Concrete garden planters, DIY paving, and outdoor furniture can all be made at home. Designs can be former-free and rustic – such as this table and chair: hsmag.cc/TableAndChair – but in most cases, you'll first need to make some form of mould or former that you subsequently remove. This wiki on making a concrete stool is an ideal entry-level concrete furniture build: hsmag.cc/ConcreteStool.

If you're a woodworker who wants to incorporate concrete into your repertoire, DIY Pete has a great concrete-topped table to start you off: hsmag.cc/ConcreteTable.

LEAF-CAST CONCRETE LAMP

Project Maker
MADE BY BARB

Project Link
hsmag.cc/ConcreteLamp

Technical illustrator, and avowed concrete fan, Barbara has a home and garden filled with her hardy creations, most striking of which are the oversized heads used for planters and her half-hidden cast concrete hooded figures.

Her leaf-based lamps formed around Pringles tubes are height-adjustable and need vinyl tubing to cover and protect their cables from the concrete. Mesh wrapped around the crisp can helps the concrete adhere. The underside of leaves are used since the veins are much more prominent than on the leaf's front. Apply leaves to one side of the tube-can former

"MESH WRAPPED AROUND THE CRISP CAN HELPS THE CONCRETE ADHERE"

and let it dry before starting on the other side. Once arranged on the tube former, they should be left in place until the concrete sets but hasn't yet caused the leaves to dry out, after which the leaf can be peeled off. Apply a very thin layer of paint, followed by a layer of matte medium. →



Right ■ Concrete can be used for delicate structures as well as large-scale building

CONCRETE FIREPIT

Project Maker
BEN UYEDA
AKA
HOMEMADE MODERN

Project Link
hsmag.cc/ConcreteFirePit

Concrete is a fashionable material for use in kitchens and dining rooms, and has the durability of marble and quartz without the ethical mining issues. HomeMade Modern offers details of how to create sharp-lined tabletops, side tables, and chairs befitting of the stripped-

"HE GIVES FULL DETAILS OF HOW TO ENSURE THE GROUNDWORK IS DONE"

back industrial look. Maker Ben's utilitarian DIY firepit features alongside other similar projects on Instructables, where he gives full details of how to ensure the groundwork is done to ensure a safe foundation on which to position the outdoor cauldron (hsmag.cc/ModernFirePit). Such details are lacking from the fancier firepit design featured on his own website, where you can access design files for 3D-printable tessellating concrete tiles that you assemble to surround your fire. You need to print all three sizes of the 3D concrete mould in order for the design to work.



Right ♦

Multiple small casts mean you don't have to deal with large amounts of concrete to get a large structure

CONCRETE CHESS

With summer on its way, it's handy to have some al fresco entertainment options in the garden.

Maker Belsey was keen to design her own chess pieces for this impressive chequerboard setup, which features figures that are "proud to be made of concrete" and is available year round for community garden visitors' use. Special cement pigment dye was mixed into the concrete for the black chess pieces. Most of the pieces are formed from concrete poured into empty drinks containers – the king's crown is the base from a soft drinks bottle – while ball pit balls provide moulds for the pawns. Knights and bishops were made from distinctive geometric shapes cut from polystyrene. Belsey includes templates for these on her website (hsmag.cc/Belsey).

She warns that the creation process involves plenty of patience. Although the pieces may seem firm after a few hours, the larger pieces need several weeks to fully cure so that they are robust enough for long-term use.

Project Maker
BELSEY

Project Link
hsmag.cc/ConcreteChessSet



Left ♦
Next step: add some black squares

OUTDOOR AQUARIUM

Arrmed with some old towels, bricks, and plenty of cement and sand, this maker set to work turning a corner of the backyard into a fishpond with transparent sides (more like an outdoor aquarium). The towels were soaked in

water, and then immersed in a concrete mix before being draped over a stool so they would acquire a distinctive, organic-looking form. Once dried stiff, the towels were placed at the corners of the in-progress fishpond and the rest of the structure built around it, with bricks forming a low wall with window effect on the front edge, into which Plexiglas was lowered. This means the fish can be viewed from a wider set of angles. Concrete was slathered over the remaining

Project Maker
GARDENING IDEAS

Project Link
hsmag.cc/OutdoorAquarium



"TOWELS WERE SOAKED IN WATER, AND THEN IMMERSSED IN A CONCRETE MIX"

structure to hide the construction fabric, and the structure painted, before aquatic plants and a water pump were added and fish were introduced. □

Left ♦
Fish deserve interesting architecture as well as humans

IN THE WORKSHOP: Sublimation printing

By Ben Everard

Transfer prints onto mugs and PCBs



Right ■
The same PCB is used for both left and right wing

A

sublimation printer looks and works a lot like an ordinary inkjet printer. You load paper into it (although it does have to be proper sublimation paper), plug it into your computer, and press

print. So far, so ordinary.

The word sublimation means to turn from a solid into a gas without being a liquid in between. This stage – the sublimation – is what happens after you've printed onto paper. It means you can turn the, now solid, ink into gas and transfer it onto other objects.

When we say other objects – there's a wide range of stuff that you can transfer sublimation prints onto, but not everything. It has to have a coating that will accept the ink. In most cases, this means buying a pre-prepared 'blank', although you can get laminates and varnishes that will let you coat the surface yourself.

Once you have the printed sheet and the object to impress the image on, you just need to stick the paper on with heat-resistant tape (aka Kapton tape), and heat-press it on. This is another limitation on the

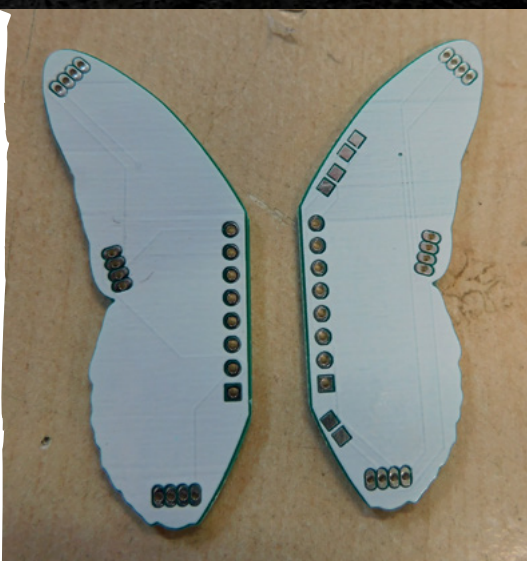
You can get heat-presses that can make custom shapes

range of objects that you can transfer images onto – you have to be able to heat-press them.

For flat surfaces, it's pretty easy with a standard heat-press. You can also get cylindrical presses for mugs. Beyond that, it's a bit challenging. For industrial uses, you can get heat-presses that can make custom shapes, but that's probably a little excessive for hobbyist use.

You have to be able to heat the whole lot up to about 200°C, so it's not ridiculously high temperatures. We have heard of people having success using convection ovens and heat-shrink to hold the designs in place, but we've not tried this, and it's probably best not to do it in an oven that's also used for food.

We've been testing out a range of different materials and blanks. Mugs, jigsaws, photo-slates, →



Left ♦
Plain white silkscreen covers the whole of the PCB

Below ♦
A bit of Kapton tape holds the PCBs in place while I heat-press on the image



It's helpful
to me if a
process is a
little tricky

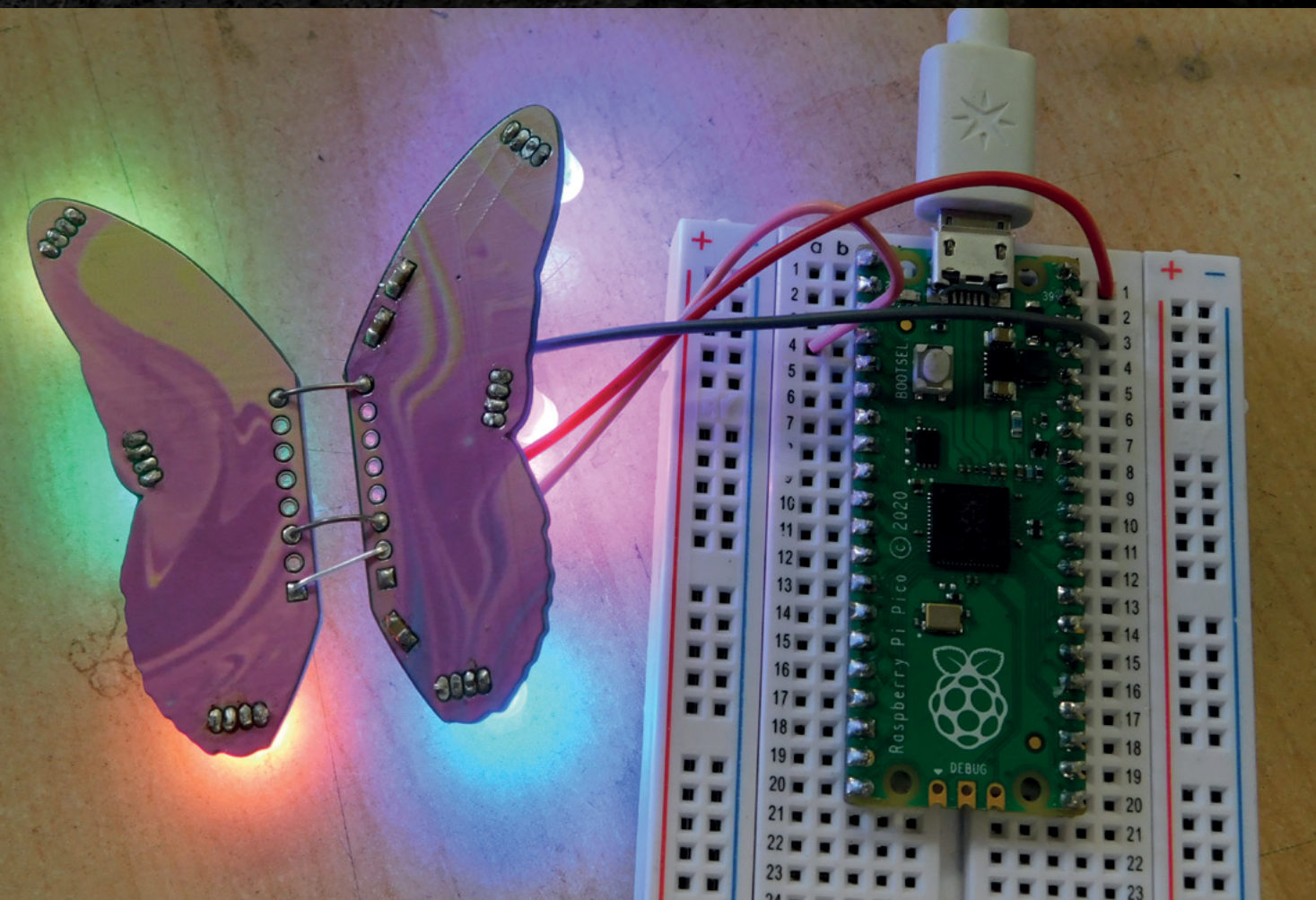
Below
I've made mugs for
our sister magazines

(slates with a flat side that has a layer of sublimation-friendly coating), and fabric.

It's helpful to me if a process is a little tricky or has a few gotchas. This gives me something to write about in the article. The problem is, everything went very smoothly. In fact, the only problem we had in the whole process was a mug heat-press that just kept beeping at us, but reading the manual and working out what button to press sorted that out.

Different materials require slightly different temperatures, and can require radically different times. As far as I can tell, heavy things like ceramics that take a long time to heat up require longer than lighter things, like jigsaw blanks. However, you should be able to get the right time from wherever you get your blanks, if not, you'll just have to experiment for a bit. About 60 seconds at 190°C was the quickest we





Above ♦
The butterfly's
LEDs can light up
any colour

did (jigsaws), and 240 seconds at 205°C was the longest (slate blanks). If you get a sepia tint, leaving your images looking like they are a few hundred years old, then you've probably overheated the ink.

As a final test, we tried sublimating ink onto a PCB, just to see what happens. The initial results were really good. While the ink did not have a great deal of effect on the FR4, it stuck extremely well to silk-screened areas.

This meant that we couldn't just sublimate print onto any PCB we wanted – we had to design specific PCBs covered in silkscreen to take the ink. I've always wanted to make a butterfly PCB. Long-term readers with a very good memory may remember that HackSpace mag, issue 1, featured a butterfly PCB from the now-defunct Boldport Club. In that, they'd used two different silkscreen colours to create the decorative effect but, in mine, I could just print whatever I wanted.

This was just a test, so I wanted something that would look good, but not take too long to design. I settled for a circuit that linked up three through-hole

RGB LEDs. The design was one wing, but I could just flip it over to create the other wing.

These came back a week or so later, and it was time to try them out. I didn't have a specific pattern in mind (this was still a test, so I didn't want to spend ages on it before knowing if it'd work), so I printed off a swirling colour image that I found online, stuck a couple of wings to it, crossed my fingers, and started the heat-press.

The result isn't quite perfect. The image doesn't quite have the detail that I've been able to get on some surfaces. I'm not yet sure if that's a result of the inherent properties of the silkscreen, or if I just haven't yet dialled in the settings (the PCBs only arrived a few days before we went to press, so I haven't been able to properly test the process out yet). So far, it seems like the ink is very prone to overheating with PCBs. The best results that I have had are at 180°C for 60 seconds, but it's still early days.

This is definitely a ripe area for experimentation if you're interested in making artistic PCBs. ▣



The **MagPi**

HackSpace
TECHNOLOGY IN YOUR HANDS

CustomPC

3 ISSUES FOR £10



FREE BOOK



hsmag.cc/hsbook

Subscribe to The MagPi, HackSpace magazine, or Custom PC. Your first three issues for £10, then our great value rolling subscription afterwards. Includes a free voucher for one of five fantastic books at store.rpiexpress.cc/collections/latest-bookazines
UK only. Free delivery on everything.

HackSpace
TECHNOLOGY IN YOUR HANDS

FORGE

HACK | MAKE | BUILD | CREATE

Improve your skills, learn something new, or just have fun tinkering – we hope you enjoy these hand-picked projects

PG
74

FIREFLY

Get started with Raspberry Pi Pico and MicroPython

PG
78



WEB SENSORS MADE EASY

Connect Raspberry Pi sensors to the web with barely any code

PG
82

POLYHEDRONS

3D shapes from 2D paper

PG
68

SCHOOL OF MAKING

Start your journey to craftsmanship with these essential skills

68 Dithering NeoPixels

PG
88

LASER AIRFLOW

Sucking and blowing your way to better cuts



PG
94

ROCKET GLIDER

Shoot your way up and float back down

CircuitPython sequences and adding brightness to NeoPixels

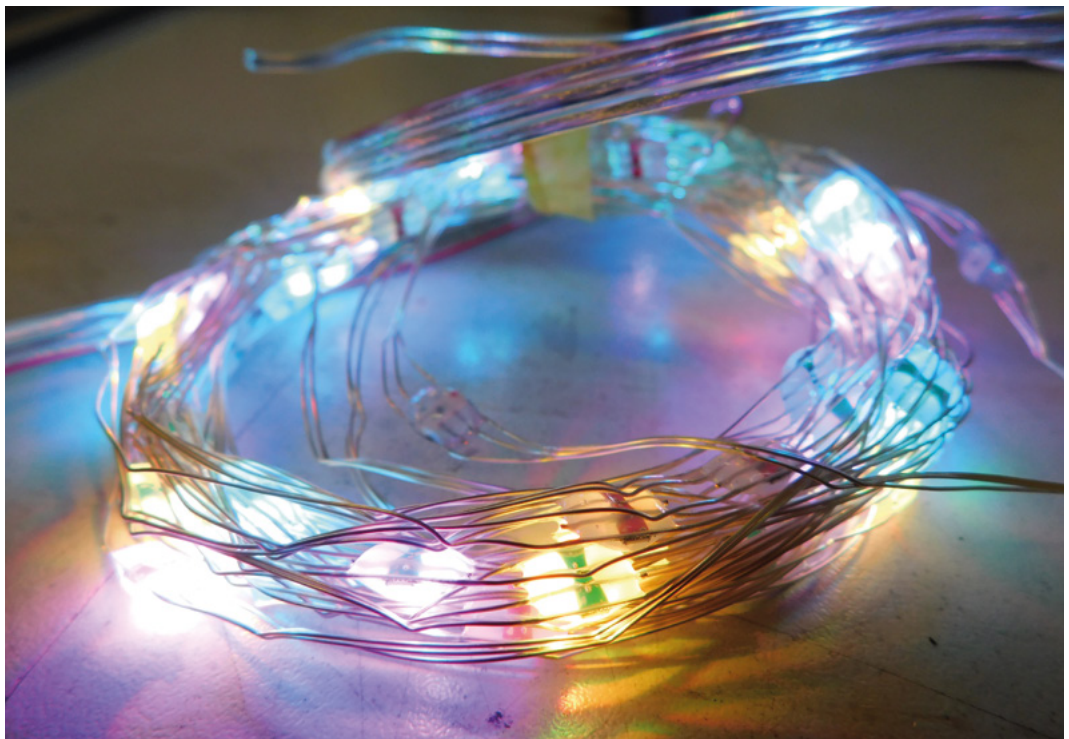
Learn to use Python's dunder methods to add new features to your classes



Ben Everard

 @ben_everard

Ben's house is slowly being taken over by 3D printers. He plans to solve this by printing an extension, once he gets enough printers.



Above  WS2812B LEDs are a great way of adding a bit of interest to your project, or an interesting light pattern could be a project in itself

In CircuitPython, PIO StateMachines have a new feature: `background_write`. This is a way of creating a set of data that will then be sent to a PIO program in the background (either once or in a loop). This means that, if you've got a lot of data to send (or need to send it continuously), you don't have to wait around for it to finish sending. As soon as you issue this instruction, the system will start the transfer, and then return control of the processor back to the program so your code will continue to run while the data

transfers. It's not as fully featured as the DMA control that you can get if you program in C, but it is easy to use and works well in many use-cases.

Let's take a look at one great application for this. Regular readers will know that we're big fans of WS2812B LEDs – sometimes known as NeoPixels. However, one problem is that although they have 24-bit colour, there is no gamma correction. This means that the 8 bits per colour channel is spread evenly through the colour range, and there are very few brightness levels available at the low end.

There is a way around this – dithering. Basically, this means flicking between different brightness levels very quickly.

Let's take a look at how to do this. Along the way we'll look at Python sequences – this is the mechanism that lets us create list-like objects.

If you've used NeoPixels with CircuitPython before, you've probably noticed that you can use a NeoPixel object like a list. For example, to set the third pixel red, you can use code like:

```
pixel[2] = (100,0,0)
```

This works even though pixel is a NeoPixel object and not a list. This neat little feature of Python isn't available in many other languages. It lets us create sequences of things that work like lists because semantically they are like lists even though, at a syntax level, they're not lists. We can do this because the Python interpreter will pass this through to a special method.

In Python (and therefore CircuitPython), this particular syntax is akin to:

```
pixel.__setitem__(2,(100,0,0))
```

Dunders (double unders before and after) are special methods that are invoked by the Python interpreter at particular times. The most common one is `__init__` that you will come across if you've ever created a class, but there are a few. As well as `__setitem__`, we'll use `__len__`, which is called whenever a programmer uses `len()`, to determine the length of our LED string.

FLASHING LIGHTS

Let's take a step back, though, and look at how we send data to the LEDs. This is done using a PIO program:

```
_program = Program(
    """
    .side_set 1 opt
    .wrap_target
```

__GETITEM__

Typically, a sequence would have a `__getitem__` dunder which allows you to query a point in the sequence. However, in our pixels, it's not quite this simple because we're not actually storing the values that a program sets. We only have the various different values. In theory, you could work backwards to recreate the level, but that's beyond the scope of this article.

```
33 class DitheredNeoPixel:
34     def __init__(self, pin, number, extra_bit_depth, order=None, bpp=3,):
35         self.byte_count = bpp * number
36         self.bit_count = self.byte_count * 8
37         self.padding_count = -self.byte_count % 4
38         self.bpp = bpp
39         self.extra_bit_depth = extra_bit_depth
40
41         self.pixels = number
42
43         self.pix_sm = StateMachine(
44             _program.assembled,
45             auto_pull=False,
46             first_sideset_pin=pin,
47             out_shift_right=False,
48             pull_threshold=32,
49             frequency=12_800_000,
50             **_program.pio_kwargs,
51         )
52
53         # backwards, so that dma byteswap corrects it!
54         header = struct.pack(">L", self.bit_count - 1)
55         trailer = b"\0" * self.padding_count + struct.pack(">L", 3840)
56
57         self.buf = bytearray((8+self.padding_count+self.byte_count)+self.extra_bit_depth)
58
```

```
pull block      side 0
out y, 32        side 0      ; get count of
NeoPixel bits

bitloop:
pull ifempty     side 0      ; drive low
out x 1          side 0 [5]
jmp !x do_zero   side 1 [3] ; drive high
and branch depending on bit val
jmp y--, bitloop side 1 [4] ; drive high
for a one (long pulse)
jmp end_sequence side 0      ; sequence is
over

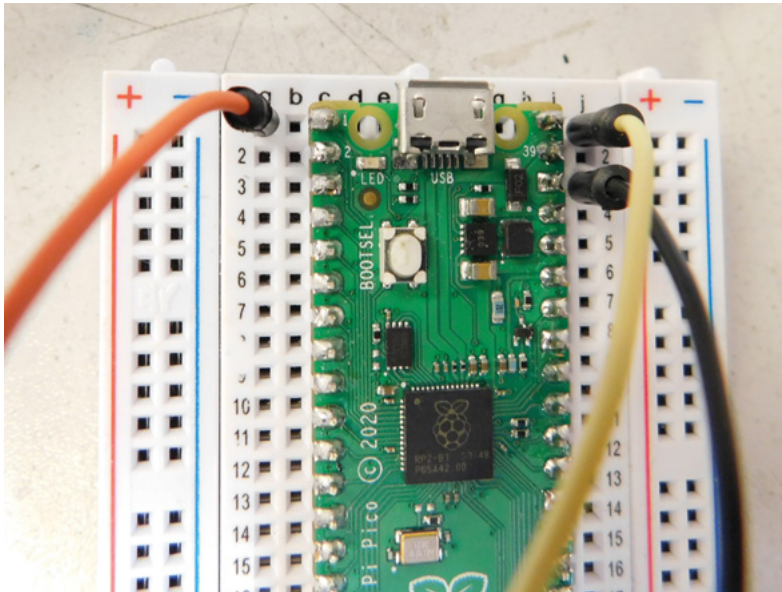
do_zero:
jmp y--, bitloop side 0 [4] ; drive low
for a zero (short pulse)

end_sequence:
pull block      side 0      ; get fresh
delay value
out y, 32        side 0      ; get delay
count
wait_reset:
jmp y--, wait_reset side 0    ; wait until
delay elapses
.wrap
"""
)
```

Above ♦
If you've not used CircuitPython before, you can get started at learn.adafruit.com/welcome-to-circuitpython

This isn't really a tutorial about writing PIO programs – we've covered this before – so we'll only look at this briefly.

PIO programs read in data from the transmit FIFO. This program first reads in 4 bytes and loads them into the y scratch register. It uses these a bit like the counter in a `for` loop to count the number of bits of data to send (typically, this will be 24 times the number of pixels with eight values each for red, green, and blue). →



For each bit it sends out, it will send a single pulse which will either be high for 2/3rds and low for 1/3rd (a logical 1), or high for 1/3rd and low for 2/3rds (logical 0). The processor sends out the values for each LED in turn.

After the allotted number of bits have been sent, the protocol dictates that the data channel must be held low (i.e. at 0V) for a set amount of time to reset the strip. This is done by loading in another 4 bits into the y scratch register and just running an empty loop that many times.

Once it reaches the bottom of the program, this loops back around to the start, and loads 4 bytes into the y scratch register and then sends out the bits again. The LEDs will stay at their previous colour until a new colour is sent to them, so in normal operation, you don't need to continuously send out data. However, we want to flick through different brightness levels as quickly as possible.

BITSTREAM

Our data stream needs to include the bit counts and the timing count at the end, as well as actual data for the LEDs. So, let's take a look at how this will actually work.

We want to create 'sub-levels' of brightness. That means that if the LED can have a brightness of, for example, between 5 and 6, we want to create steps between these two. We can do this by flicking backwards and forwards between the two levels very quickly. We'll create a bitstream with more than one set of data so the whole thing will be sent in one go. First, it'll set the LEDs to one brightness level, then another, and then another and so on. It'll do this faster than our eyes can see, so this will create intermediate brightness levels.

For example, if we flick back and forwards between level 5 and 6, we'll get a brightness that appears about halfway between them. If it spends longer at level 5 than 6, then it'll seem closer to level 5 than 6, so we can create multiple sub-levels of brightness.

The key parts of our code need to do two main things. First, we need to create a byte array with the bit counts and pause counts in place. We then need to work out a way of deciding what levels to give the LEDs to get the levels of brightness we want.

We create our blank buffer inside our class's `__init__` method:

```
class DitheredNeopixel:
    def __init__(self, pin, number, extra_bit_
depth, order=None, bpp=3,):
        self.byte_count = bpp * number
        self.bit_count = self.byte_count * 8
        self.padding_count = -self.byte_count % 4
        self.bpp = bpp
        self.extra_bit_depth = extra_bit_depth

        self.pixels = number

        self.pix_sm = StateMachine(
            _program.assembled,
            auto_pull=False,
            first_sideset_pin=pin,
            out_shift_right=False,
            pull_threshold=32,
            frequency=12_800_000,
            **_program.pio_kwargs,
        )

        # backwards, so that dma byteswap corrects
it!

        header = struct.pack(">L", self.bit_count
- 1)

        trailer = b"\0" * self.padding_count +
struct.pack(">L", 3840)

        self.buf = bytearray((8+self.padding_
count+self.byte_count)*self.extra_bit_depth)

        offset = 0
        for i in range(self.extra_bit_depth):
            for j in range(4):
                self.buf[offset+j] = header[j]
            for j in range(self.padding_count+4):
                self.buf[offset+self.byte_
count+4+j] = trailer[j]
            offset = offset+self.byte_
count+8+self.padding_count
```

Above ♦
The wiring is simple:
you just need 5V
output (from VBUS),
ground, and signal.
The connections
should be marked on
your LED strip

The parameter **extra_bit_depth** is used to hold the number of sequences of the data we'll hold. This isn't actually **bit_depth** since if we hold four copies of the data, that's not equivalent to four more bits, it's only equivalent to two bits, but this doesn't affect the running of the code, it's just terminology.

The **buf** variable is a buffer that holds all the data for all the runs. However, we have to have a header and a trailer for each run to set up the bit count and pause correctly.

As we mentioned at the start, we need the **__setitem__** method so that we can use Python's sequence syntax to update the list. This method has to take three parameters – **self**, the position in the sequence, and the value that we want to assign to that position in the sequence.

Our LEDs take three values – one each for red, blue, and green – however, we can get around this by taking a tuple. This means that we assign a colour to a pixel with:

```
pixel[i] = (10,20,30)
```

The code for this is:

```
def __setitem__(self, number, colour):
    for i in range(self.bpp):
        #find closest 8 bit value
```

```
        error = 0
        for byte in range(self.extra_bit_
depth):
            val = (colour[i] + error) >>
self.extra_bit_depth

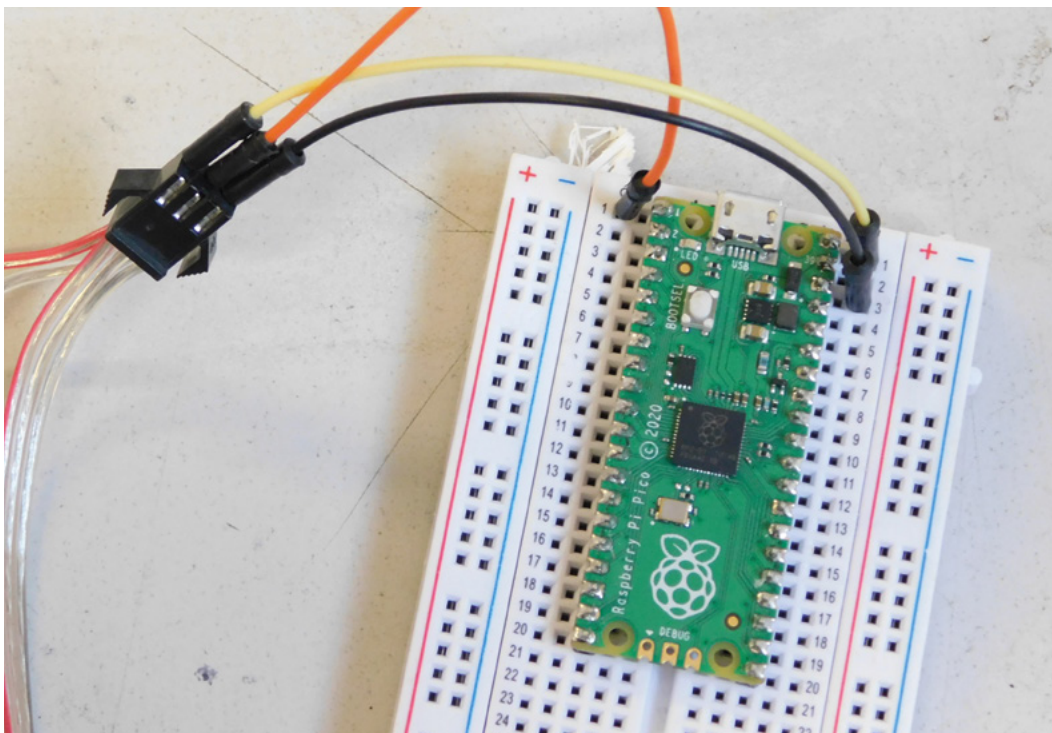
            self.buf[(self.byte_count+self.
padding_count+8)*byte + 4 + number*3 + i] = val
            error = (colour[i]+error) - (val
<< self.extra_bit_depth)
```

This uses a dithering algorithm that keeps track of the difference between what the LED should show and what it has shown with the different levels up to this point.

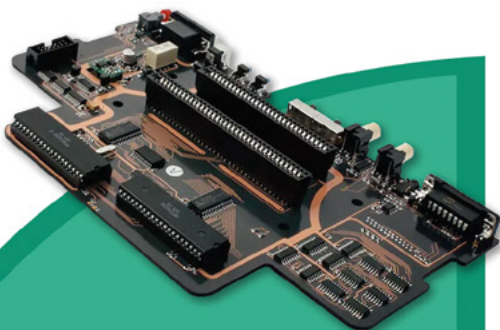
We now have a class that we can use more or less like the classic **NeoPixel** class, except that we can set brightnesses beyond the original 8 bits per channel.

Of course, in this example, we haven't added any error checking, so it's probably possible to get it into some strange states! You might also find that some LEDs have red, green, and blue in a different order. You can swap this around in your code.

You can grab all the code for this project from **hsmag.cc/DitherNeoPixel**. It's unpolished, but you can tweak it to your heart's content to make it work in your project. □

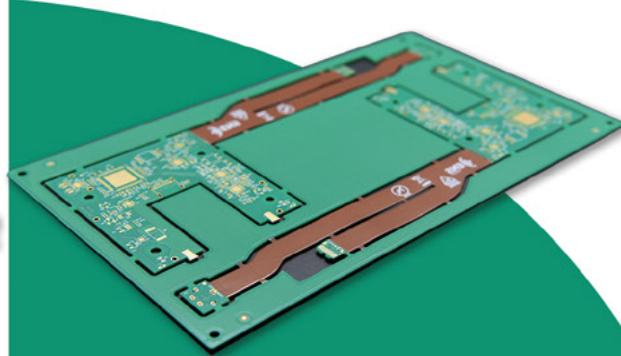


Left ♦
We've used a breadboard to make the connections easier, but you can also solder them in place or use headers



- High-Quality PCBA with component sourcing
- Start from only **\$30**
- Free stencil & Free shipping all over the world

Turnkey PCB Assembly



- Instant quote online & User-friendly web interface
- Fast turnaround in **24 hours**
- Make your unique and high-end products the easy way

Advanced PCB Fabrication

**Quality
Guaranteed**

3D Printing & CNC Machining

- Custom metal & plastic parts for **only \$25**
- On-demand production and rapid prototyping in **as fast as 1 day** leading the digital manufacturing revolution.



More Custom Options

- Professional Flex/Rigid-Flex PCB
- New soldermask colors: **Pink, Gray, Orange and Transparent**
- Perfectly implement your idea with different types of PCB



More information:
www.PCBWay.com

Email:
service@pcbway.com

Phone:
(0571) 8531 7603

THE OFFICIAL Raspberry Pi Beginner's Guide

The only guide you
need to get started
with Raspberry Pi

Inside:

- Learn how to set up your Raspberry Pi, install an operating system, and start using it
- Follow step-by-step guides to code your own animations and games, using both the Scratch 3 and Python languages
- Create amazing projects by connecting electronic components to Raspberry Pi's GPIO pins

Plus much, much more!

£10 with FREE
worldwide delivery



Buy online: magpi.cc/BGbook

LED firefly

Learn MicroPython and create your own electronic pet



Rebecca Franks

 @FranksberryPi

Rebecca is a learning manager at the Raspberry Pi Foundation. She creates computing resources for teachers and young digital makers. She has over 15 years experience teaching computing.

In this project, you'll use a Raspberry Pi Pico to make an LED firefly that flashes in a particular pattern, just like fireflies in nature, and connect a switch to control the light.

To complete this project you will need a few common components.

You can purchase all the required hardware for this project, and the other projects in this path, from most hobbyist electronics stores, but there's a kit specifically for the Pico learning path (see box) available from Pimoroni (hsmag.cc/PicoIntroKit).

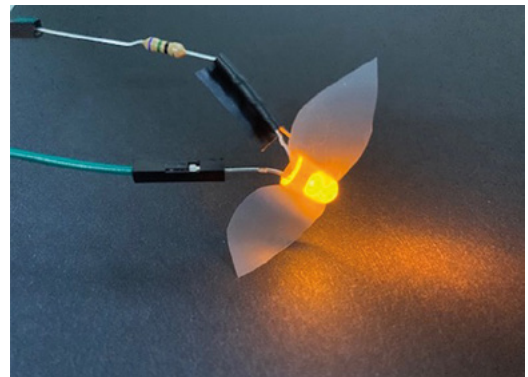
Before diving into the project, you'll need to attach a resistor and jumper wires to the LED, as shown in the information box on this project page: hsmag.cc/PicoLEDs.

We'll be using Thonny to program Pico. If you're using a Raspberry Pi computer, you should already have Thonny installed. If you're using a different machine, you can download and install it from thonny.org.

Now we've got all the bits for our project, let's get started. First, we need to set up Pico to run MicroPython.

Connect the small end of your USB cable to the Raspberry Pi Pico, then connect the other end to your computer, laptop, or Raspberry Pi.

Open the Thonny editor, and look at the text in the bottom right-hand corner of the Thonny editor. It will show you the version of Python that is being used. If it does not say 'MicroPython (Raspberry Pi



Pico)', then click on the text and select 'MicroPython (Raspberry Pi Pico)'. If you have never used MicroPython on your Raspberry Pi Pico, then Thonny will prompt you to add the MicroPython firmware. Click Install.

To complete the projects in this path, you will need to install the picozero library as a Thonny package. In Thonny, choose Tools > Manage packages, then in the pop-up 'Manage packages for Raspberry Pi Pico' window, type 'picozero' and click Search on PyPi. Click on picozero in the search results, and click on Install. When installation has completed, close the package window, then exit and reopen Thonny.

If you have difficulties installing the picozero library in Thonny, you can download the library file and save it to your Raspberry Pi Pico.

Right 
The tape isn't needed, but does add a little graphical flair

Now we'll light the tiny LED that sits on the top of your Raspberry Pi Pico. This will check that your Raspberry Pi Pico is set up correctly.

Create a new file in Thonny by clicking File > New in the top menu bar. An empty workspace will open.

The picozero library allows you to program electronics components that are attached to a Raspberry Pi Pico. At the top of your code, you will

//

The LED will only
light for one second,
so make sure you
are watching

//

need to import the items that you need from the picozero library.

Type the following code into the main editor pane in Thonny:

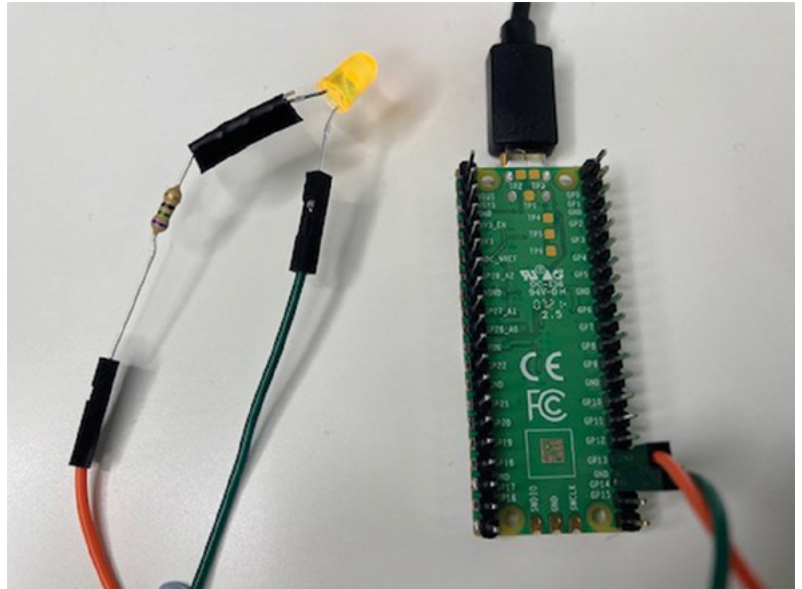
```
from picozero import pico_led

pico_led.on()
```

Choose File > Save As. Thonny will ask whether you want to save the file on This computer or the Raspberry Pi Pico. Choose This computer to save your code to your computer.

PICO PATH

The Raspberry Pi Pico is a low-cost microcontroller device. The Raspberry Pi Foundation has developed a new path of projects to allow complete beginners to make engaging, interactive gadgets with a small kit list of low-cost materials, including a Raspberry Pi Pico, RGB LEDs, passive buzzers, and everyday craft supplies. The projects use MicroPython with the new 'picozero' beginner library that allows you to write code such as `led.pulse()` to pulse an LED. You don't need a breadboard, and you don't need to solder components (unless you want to). The projects have been designed for young creators from age 9 to 13, but they are also suitable for older makers who haven't found a way to get started with microcontrollers. The path of six projects starts with three step-by-step projects to create an LED firefly, a party popper, and a beating heart. The next two projects guide you through designing and making a mood indicator and a sound machine. Finally, you are encouraged to apply your skills to invent and make a sensory gadget. The projects are ideal for learners at home or in clubs and maker spaces. Check out the Introduction to Raspberry Pi Pico path at hsmag.cc/PicoProjects.



Choose a location on your computer such as your 'Documents' folder. Name your file **firefly.py**.

Now let's test the code. Thonny has a green play button with a small white triangle inside it. Pressing this button allows you to run your code. Press this button to run the code.

Check that the small LED on the Raspberry Pi Pico turns on.

The LED will stay on until you write code to turn it off, or you unplug the Raspberry Pi Pico.

Above ♦

By pre-soldering your LEDs to resistors, you can build circuits with jumpers

ON AND OFF

The sleep module allows you to pause your code for a specified number of seconds. Update your code to the below to turn the LED on, wait one second, and then turn it off.

```
from picozero import pico_led
from time import sleep

pico_led.on()
sleep(1)
pico_led.off()
```

Click the green play button. Thonny will save the file on your Raspberry Pi Pico and then run the new code.

Check that the LED turns on and then goes off again. The LED will only light for one second, so make sure you are watching. Run your code as many times as you like.

Let's now switch to an external LED that we can use for our firefly. →

YOU'LL NEED

- ♦ A Raspberry Pi Pico with pin headers soldered on
- ♦ A data USB-A to micro USB cable
- ♦ Yellow LED (or any colour you prefer)
- ♦ 100Ω resistor (any resistor from 75Ω to 220Ω will work)
- ♦ Pin-socket jumper wire
- ♦ 3 × socket-socket jumper wire
- ♦ Optional: Sticky tape (invisible tape works best)

Right  Pico has a built-in LED in the top right-hand corner

Below  The LED connects between GP13 and Ground



In this step, you will use jumper wires to connect an LED with a resistor to your Raspberry Pi Pico and write code to light it.

Make sure that you have an LED connected to a resistor and two socket-to-socket jumper wires. The colour of the jumper wires does not matter – they all do the same thing.

Fireflies are usually yellow, orange, or green, but you can choose any colour you like.

A resistor controls the current that flows through a circuit. This protects the LED from burning out and will make it last longer.

Raspberry Pi Pico has 40 pins on its board. Pins allow you to connect external components to the Raspberry Pi Pico.

Explore your Raspberry Pi Pico and find the pin that is labelled GP13. You will notice that there are labels for each pin on the underneath side of the Raspberry Pi Pico.

Connect the jumper wire that is attached to the positive leg of the LED (the one with the resistor) to pin GP13. Push it until the black plastic meets the base of the header.

Connect the jumper wire that is attached to the negative leg of the LED to the GND (ground), below GP13. This completes the circuit, allowing electrical current to flow when instructed by your code.

In the last step, you used `pico_led` to light the LED on the Raspberry Pi Pico. To add your own LEDs, you need to import `LED` from `picozero`.

Add `, LED` to the end of the import list on line 1. Next, set your firefly LED to GP13 and enter the code to switch it on.

Enter the code to light your firefly:

```
from picozero import pico_led, LED
from time import sleep

pico_led.on()
sleep(1)
pico_led.off()

firefly = LED(13) # Use GP13
firefly.on()
```

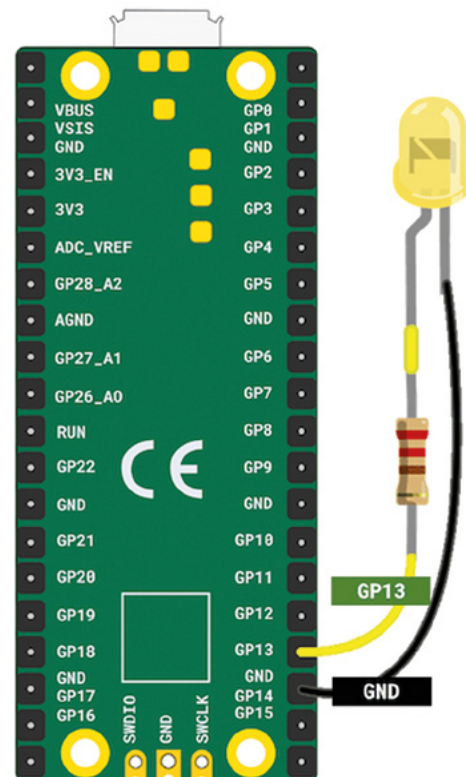
You can optionally cut some wings out of folded-over sticky tape and stick them to your LED. Invisible tape works well.

Now we've lit up the LED, let's make it blink. Change your code to make the firefly blink on and off in a `while True:` loop. The timings represent the light patterns of a real firefly.

Make sure that the code on lines 11–14 is indented.

```
firefly = LED(13) # Use GP13

while True:
    firefly.on()
    sleep(0.5)
```




```
firefly.off()
sleep(2.5)
```

USER INPUT

It's useful to be able to control when your LED firefly starts blinking and to be able to turn it off from the device.

The Raspberry Pi Pico can detect when an input is connected between GND and one of the GP pins.

The simplest kind of switch is two jumper wires that can be connected together to close the switch, or separated to open the switch. Find one pin-socket jumper wire and one socket-socket jumper wire – the colours do not matter. Connect one jumper wire to GP18 and the other to GND. It doesn't matter which jumper wire you connect to which pin.

The simplest kind of switch is two jumper wires that can be connected together to close the switch

To add switches, you need to import **Switch** from the **picozero** library.

Add, **Switch** to the end of the import list on line 1. Next, set your switch to GP18:

```
from picozero import pico_led, LED, Switch
from time import sleep
```

```
pico_led.on()
sleep(1)
pico_led.off()
```

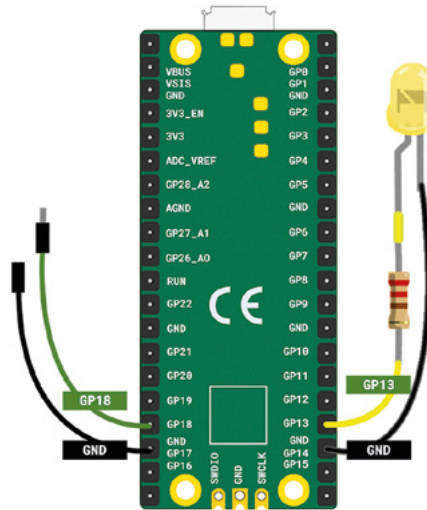
```
firefly = LED(13) # Use GP13
switch = Switch(18) # Use GP18
```

When you connect the two jumper wires together, this completes a circuit and allows the Raspberry Pi Pico to detect that the switch is closed.

Add code to check if your switch is `is_closed` (the jumpers are connected) and only blink the firefly if it is closed:

```
switch = Switch(18) # Use GP18

while True:
    if switch.is_closed: # Wires are connected
        firefly.on()
        sleep(0.5) # Stay on for half a second
        firefly.off()
```



Left ♦ This is all you need for your firefly

```
sleep(2.5) # Stay off for 2.5 seconds
else: # Wires are not connected
    firefly.off()
    sleep(0.1) # Small delay
```

Make sure the jumper wires are not connected, then run your code.

Now connect the jumper wires together. The firefly should start to blink.

Disconnecting the jumper wires will not stop power to the LED firefly immediately. The firefly only turns off when the `firefly.off()` code runs.

GOING FURTHER

If you have time, upgrade your firefly project. You could use your project as a room decoration to remind you of nature when you are indoors.

You could upgrade your project by:

- Adding another LED firefly and making it flash in the same pattern.
- Changing the flash pattern – you could research the flash patterns of fireflies in nature.
- Put your LED firefly in a jar. You could even use conductive thread to hang an LED from the lid. □

LED

LED stands for light-emitting diode. It uses electroluminescence, which is where a material lights up when an electrical current passes through it. An LED has two legs – a long one and a short one – and must be connected the right way around. The long leg is positive (+) and the short one is the negative (-). Another way to check if a leg is positive or negative is to use your finger to find the flat side of the LED bulb. The flat side is on the same side as the negative leg.

CDP Studio: Sense HAT



Phil King

WRITER

Long-time contributor to *The MagPi*, Phil is a freelance writer and editor with a focus on technology.

@philkingeditor

Use the low-code, block-based environment of CDP Studio to read sensors on a Sense HAT

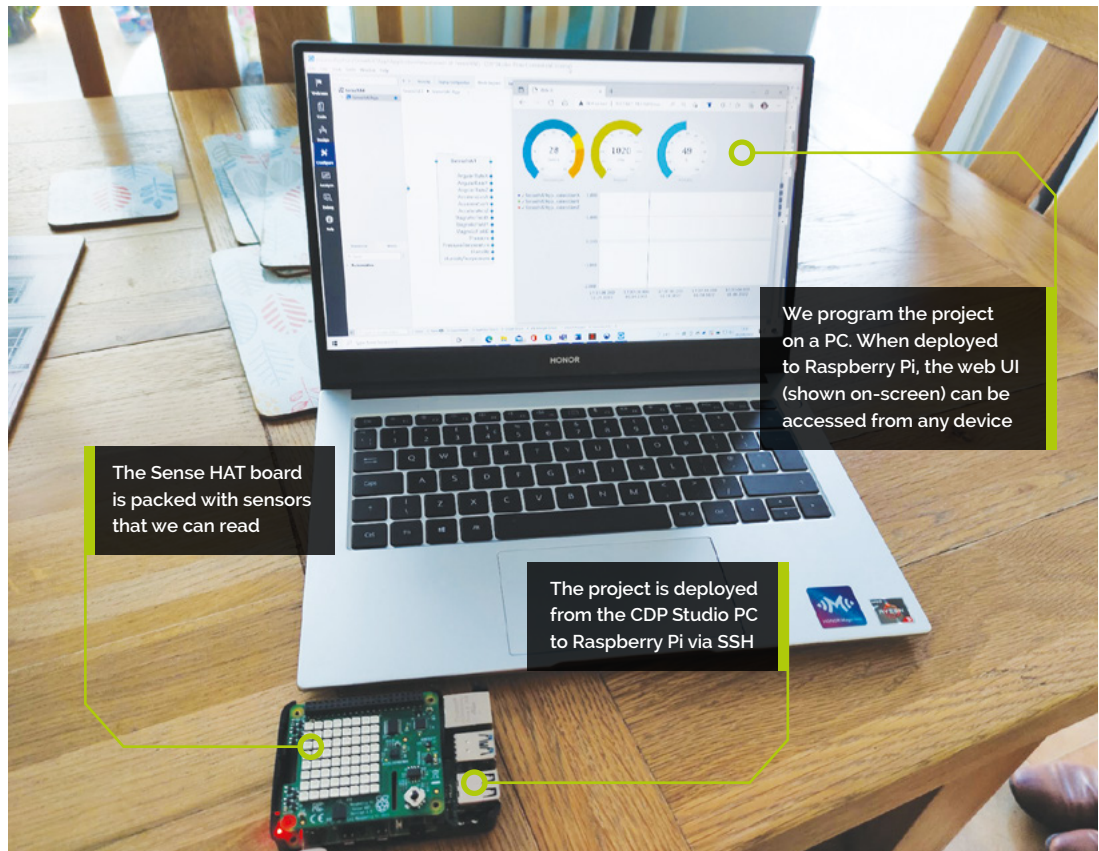
As an ‘out of the box’ software development tool, CDP Studio is used to build industrial control, automation, and edge systems. Yet it’s fairly easy to get to grips with its low- (or even no-) code programming environment, and you can deploy projects to a Raspberry Pi.

Last time, we created an app to flash LEDs connected to a Raspberry Pi in different patterns, selectable on a web GUI. In this second tutorial,

we’ll be reading the signals from the numerous sensors on a Sense HAT attached to our Raspberry Pi. Using CDP Studio’s built-in ‘SenseHAT’ recipe, this is very easy to do – programming-wise, this project only involves a single preset block in the Block Editor. We can then display the readings with a variety of widgets and graphs in a web GUI that can be viewed in a browser on any device on the network.

You’ll Need

- ▶ Linux or Windows PC
- ▶ CDP Studio
cdpstudio.com/getstarted
- ▶ Raspberry Pi
- ▶ Raspberry Pi OS (Bullseye or Legacy version)
- ▶ Sense HAT



01 Install the software

Visit cdpstudio.com/getstarted and download the free non-commercial version for Linux or Windows. During installation, make sure you select both the Raspberry Pi ARMv8 32-bit (Debian 11) and Raspberry Pi ARMv6 32-bit (Debian 10) components, along with the one already ticked for your host PC. You will then be able to deploy projects to any Raspberry Pi model, using Raspberry Pi OS Bullseye or Legacy version, by selecting the appropriate toolkit in CDP Studio.

02 Prepare Raspberry Pi

If you've followed the first tutorial, your Raspberry Pi should already be ready to use with CDP Studio. If not, you'll need to prepare it, which involves enabling SSH and modifying its `/etc/security/limits.conf` file – see magpi.cc/cdprpisetup for details.

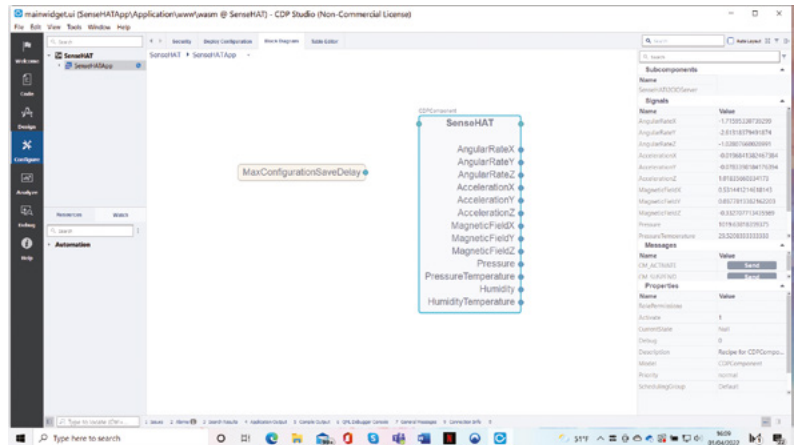
03 Start a new project

Unlike last time, there's no need to create a library for this project. So just go to File > Create New and choose CDP System. Give it a name (such as SenseHAT) and click Next, then Next again. From the Application Type drop-down, choose WebUI to automatically create a web UI for it – although if you forget to do this, you can always add one later. Now click Finish to create the system. Unless you changed it, its default app will be SenseHATapp, as shown in the hierarchy in the left panel of Configure mode.

04 Add SenseHAT component

In the Block Diagram tab, you'll see the application block. Either double-click it or click the app in the hierarchy to open it up. Don't worry about the default blocks there. In the bottom-left panel, under Resources, open the I2CIO category. Now drag the SenseHAT component there into the middle of the block diagram.

As you can see, it has outputs (on the right of the block) for all of the Sense HAT's sensors. These include angular rates and acceleration for the X, Y, Z axes of the HAT's gyroscope and accelerometer, to sense orientation and movement in three dimensions. There are also magnetic field outputs for its magnetometer. Finally there are pressure,



humidity, and two temperature outputs depending on which sensor you want to use.

▲ The preconfigured SenseHAT block provides readings from all the sensors, seen here in the right-hand panel

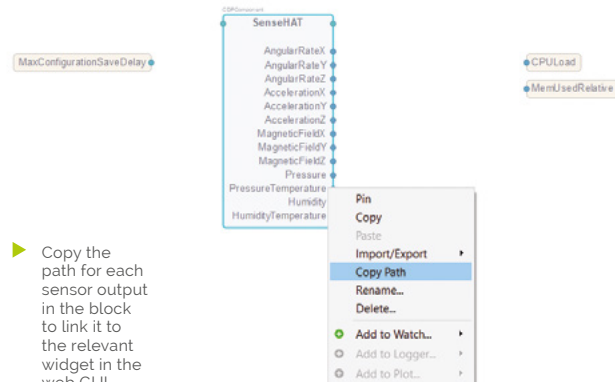
05 Deploying the program

This time, there's no need to add any other blocks to the application, nor wire anything up. The SenseHAT component will work as it is to take live readings from the HAT's sensors. Let's try it out by deploying the application.

To do so, go to the Deploy Configuration tab and select WiFi under Networks to show devices on your network. Find your Raspberry Pi by its IP address, enter pi as the Username, and click Pair, then enter its password.

Now make sure its IP address (or name if you've renamed it) is selected under Device in Applications, and that you have the relevant Toolkit selected for your version of Raspberry Pi OS: Raspberry Pi ARMv8 for Bullseye, or Raspberry Pi/Raspbian ARMv6 for Legacy.

Right-click SenseHAT in the left-panel hierarchy and choose Run & Connect to run the application on your Raspberry Pi. Wait for it to deploy. Now select the SenseHAT component in the Block Diagram and you'll see the live signals from the HAT's sensors in the right-hand panel. →

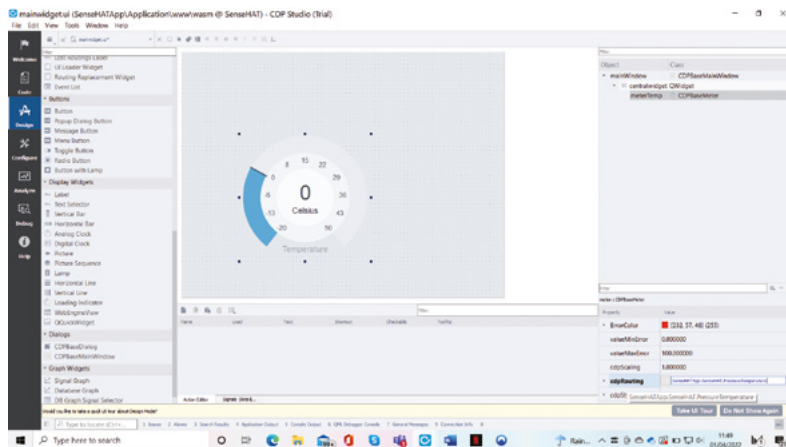


▶ Copy the path for each sensor output in the block to link it to the relevant widget in the web GUI

Top Tip

Deploy on PC

Before deploying the project on Raspberry Pi, you may want to test it by running it on the local PC, in which case the web GUI is at <http://127.0.0.1:7689/index.html>



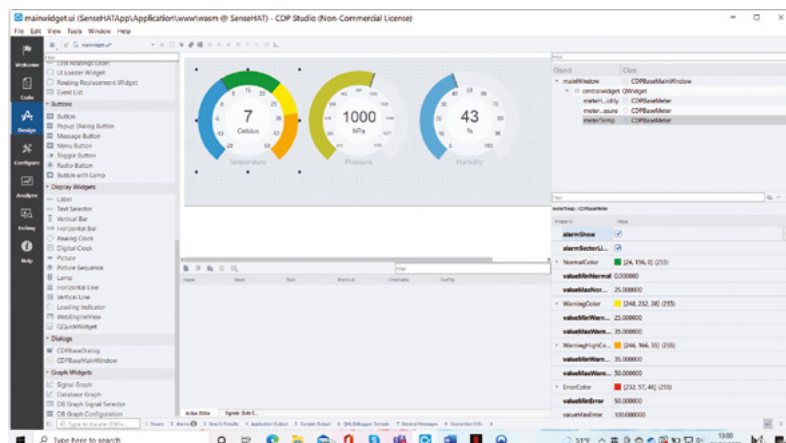
06 Design a web GUI

That's all very well, but it's not the most user-friendly way to view the sensor readings. Right-click the system or app name in the hierarchy panel and select Stop.

Got to Design mode to start designing a web GUI to show your readings. In the left panel, you'll see a variety of elements that you can add to the GUI canvas in the middle. We'll start by adding some widgets to show the temperature, pressure, and humidity readings from the Sense HAT.

We opted to use a standard Meter widget for ours, but you could use a Vertical Bar (under Display Widgets) for one or more readings if you prefer. Simply drag the widget from the left panel onto the canvas to add it. In the bottom-right properties panel, you can then alter its minimum and maximum values (in `minValue` and `maxValue`) and text labels (`textPrefix` and `textSuffix`). For our temperature meter, we set min and max values to -20 and 50, and labels to Celsius and Temperature. To send the sensor data to the meter, you will need to add the routing for it. Go back to the Block Diagram and right-click the SenseHAT output you want – we chose Pressure Temperature (the temperature reading from the pressure sensor

- ▲ Creating the web GUI in Design mode. Paste the routing for the sensor output into the meter widget's `cdpRouting` field
- ▼ You can alter the fill colours for the meters if you like, and even add coloured sectors for different value ranges



– and Copy Path. Back in Design mode, paste it into the field for `cdpRouting` at the bottom of the properties panel.

07 Test your first meter

Let's check our GUI and temperature meter are working correctly. As before, right-click the system name in the hierarchy and choose Run & Connect. Once it has been deployed, the Application Output panel at the bottom will show the URL for the web GUI. It will be your Raspberry Pi's IP address followed by **:7869/index.html** – for example, ours was **"192.168.1.112:7869/index.html"**.

Open it in a web browser and you should see the meter in your GUI showing the current temperature reading.

08 Add more meters

Stop the application from running. We'll now add meters for pressure and humidity to our web GUI. In Design mode, drag a meter over for both (or copy and paste your temperature meter). Then change the min and max values and text labels in the properties panel, as we did before.

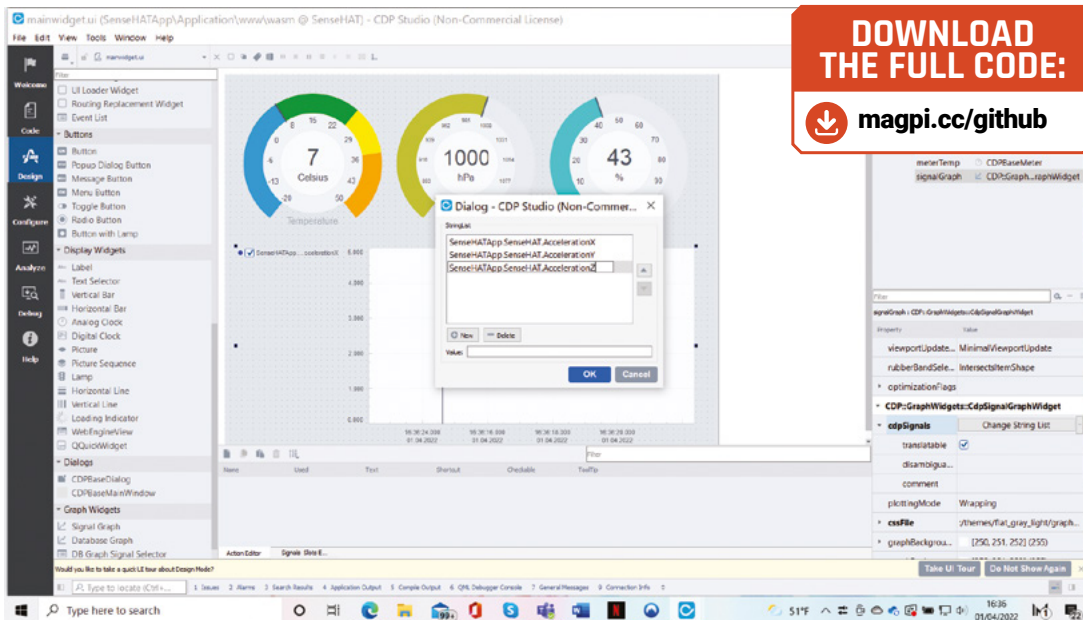
For our pressure meter, we opted for min and max values of 870 and 1100, with text labels of hPa and Pressure. For the humidity meter, the min and max values should be 0 and 100 (as it's always a percentage), while the text labels are % and Humidity.

Again, you will need to add the routing for each meter so it displays the correct sensor signal.

So right-click the relevant output (Pressure or Humidity) of the SenseHAT block in the Block Diagram, select Copy Path, and – in Design Mode – paste it into the `cdpRouting` field of the properties panel for the meter.

If you want to get creative, you can alter the `fillColor` values in the properties to change the colour of each meter's fill colour so they're not all blue. For our temperature meter, we also set min and max values for `NormalColor`, `WarningColor`, and `WarningHighColor` to allocate the colours to a range of temperatures on our meter.

When ready, run the application again and you should see all three meters in the web GUI, showing the relevant sensor readings.



- Creating a graph in Design mode. Paste the routing paths for the sensor outputs into the String List
- The finished web GUI, with meters for temperature, pressure, and humidity, plus a graph plotting the three axes of the accelerometer

09 Add a live graph

Meters and display widgets aren't the only way to display data in the web GUI. Let's add a Signal Graph to our GUI. You'll find it under Graph Widgets in the left panel of Design mode. Drag it into the GUI canvas (you may need to expand the latter) and alter its size accordingly.

We opted to show the readings from the Sense HAT's accelerometer for our graph. In the graph's properties panel, scroll down to find `cdpSignals`, then click `Change String List` next to it. This brings up a dialog where you can paste the routing for each sensor output you want to add. As before, you can obtain the routing by right-clicking the output of the Sense HAT block in the Block Diagram and selecting `Copy Path`. Our paths were `SenseHATApp.SenseHAT.AccelerationX` and so on (replacing X with Y and Z).

We also set the `yAxisMinValue` and `yAxisMaxValue` for the graph to `-2` and `2` – since we found them to be the min/max G values read from the Sense HAT's accelerometer. Of course, you could add other sensor readings to the graph if you want, or even add a second graph for something that needs a different vertical scale.

10 Test it out

Our web GUI now features meters for temperature, pressure, and humidity, along with a signal graph to show live readings for the three axes of the accelerometer. Let's try it out.

As before, Run & Connect the application to deploy it to your Raspberry Pi, then visit the web GUI URL in a web browser – don't forget to

refresh it so you see the revised GUI. Under your three meters, the graph should show the current accelerometer readings. With Raspberry Pi lying flat, the Z axis reading should be 1G (the others 0). If you tilt Raspberry Pi up to stand on its short side, the X reading will be 1; stand it on its long side and the Y value will be 1. To get higher readings, try shaking Raspberry Pi!

Taking it further

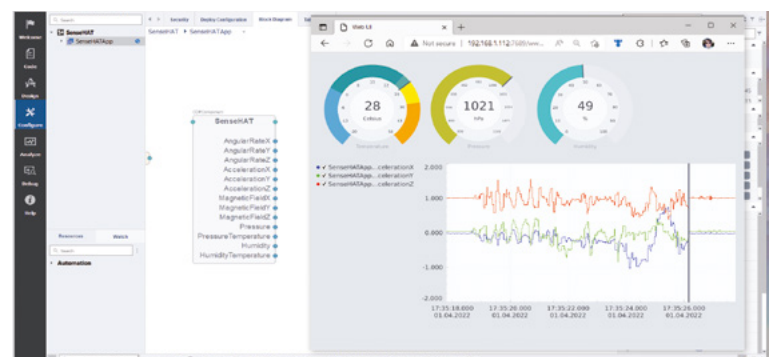
We now have a web GUI showing live readings from the Sense HAT. You could add more sensor readings to it, alter its layout, or jazz it up however you want. You could also log your sensor data using a `CDPLogger` block – see magpi.cc/cdplogger for more details – and plot it on a graph using the `Database Graph` widget in Design mode.

You're not restricted to the Sense HAT either: you could read values from sensors connected to the GPIO pins on Raspberry Pi using a `GPiO Server` block (as we used in part one). CDP Studio also features built-in support for I2C devices and the `ADS1115 ADC` – see magpi.cc/cdpizcio. □

Top Tip

Random issue

When the project has been deployed on a Windows PC, the three `AddRandom` blocks will set the same value at the same time, turning all three LEDs on/off together. When deployed to Raspberry Pi, however, each individual LED will toggle randomly.



Polyhedra

Maths doesn't have to involve solving equations, as Mike Bedford proves by showing you how to make polyhedra using card and glue



Mike Bedford

Despite loving all things digital, Mike admits to being a bit of a Luddite, vinyl records and all.

Polyhedra are three-dimensional mathematical objects, so making them as physical models might seem an obvious thing to do.

Building them was once a popular activity among those with a hobby interest in maths, and even by those with a passion for arts and crafts. But in the past couple of decades, physical models have lost support in favour of on-screen images. We wouldn't want to dismiss the appeal of computer-rendered models of polyhedra, but we think that physical models also have their appeal. After all, there's surely something rather special about an object you can admire as you hold and rotate it in your hands, so we're going to turn back the clock. If you'd like to make your own impressive polyhedral models, just collect together some sheets of coloured card and some glue, and read on.

“

There's an infinite number of polygons, depending on the lengths of their sides and their internal angles

”

FROM POLYGONS TO POLYHEDRA

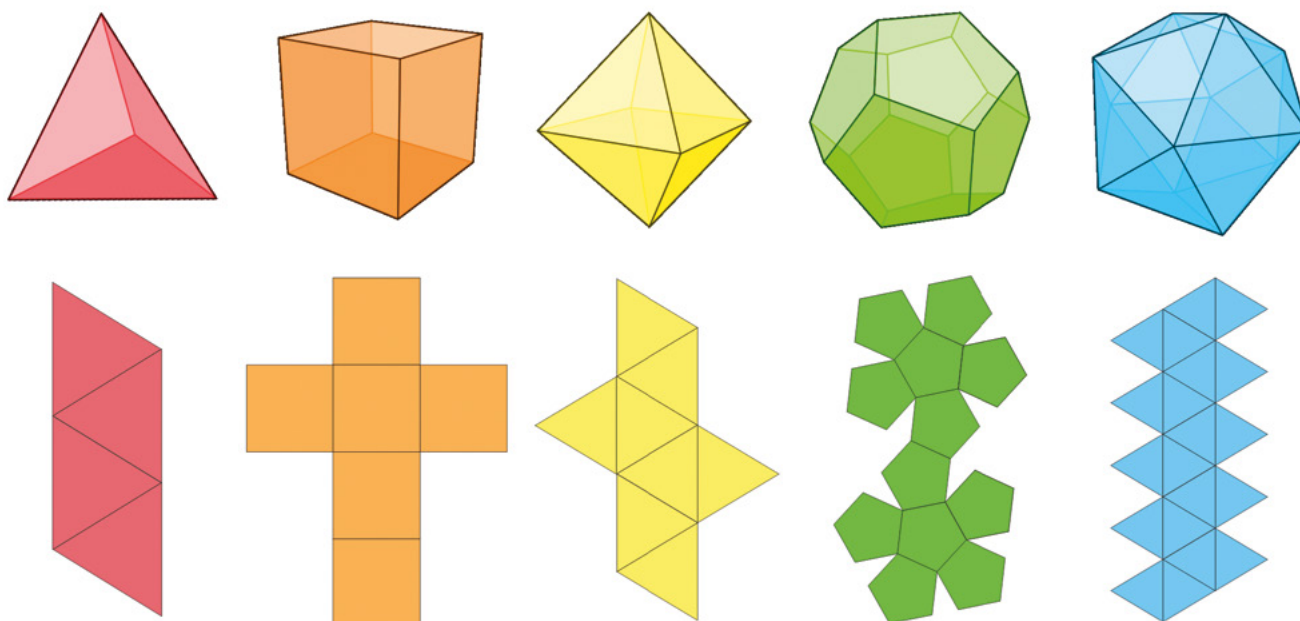
You might not have learned about polyhedra in school, but you'll surely know about polygons. A polygon is a multisided two-dimensional figure; for example, triangles, rectangles, pentagons, and hexagons. There's an infinite number of polygons, depending on the lengths of their sides and their internal angles. But we tend to be more familiar with regular polygons – those in which all the sides are the same length and all the internal angles are the same.

When we turn our attention to three-dimensional space, we find polyhedra instead of polygons. A polyhedron is a three-dimensional figure with multiple faces and their names are similar to those of polygons. So, for example, we have tetrahedra, pentahedra, hexahedra, and heptahedra. Like polygons, polyhedra can be regular; in this case, that means all the faces are identical regular polygons, and all the angles at any corner are equal. But here's an interesting thing: while there's an infinite number of regular polygons, the same isn't true of polyhedra. It might seem surprising, but there are only five regular convex polyhedra, the so-called Platonic solids, which are the cube, regular tetrahedron, regular octahedron, regular dodecahedron, and regular icosahedron. And in case you'd wondered, as we had... you probably figured from the 'poly' that a polyhedron has lots of something, and let's face it, they have lots of all sorts, but it transpires the Greek word 'hedra' means base or seat.

MAKING PLATONIC SOLIDS

Before looking at how to make a polyhedron, let's think about taking a cube like a cardboard box and unfolding it, cutting along some edges as necessary, but ensuring that it remains as a single piece. You'll end up with a flat piece of cardboard, which might be like the one in the second part of the illustration (at the top of the facing page), depending on how you unfolded it. This is called a net, and it's fairly obvious that it can be folded back into a cube. Any polyhedron can be created by folding a net, and this is the easiest way to make one.

With the Platonic solids, you could fairly easily draw the nets by hand, using a ruler and protractor. However, it's easier to find one online and print it onto card. Choose a net with tabs for gluing, because if you use a net without tabs, you'll have

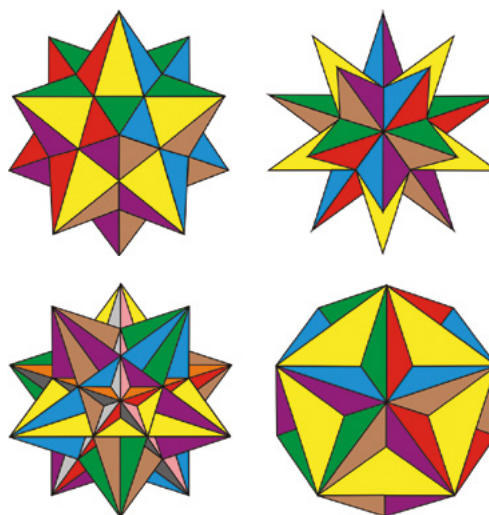


to use adhesive tape, some of which will end up on the outside of your finished polyhedron. Beware that some printable nets aren't too accurate; indeed, our dodecahedron net wasn't perfect. Once you've printed the net, cut it out and, if the short edges of some tabs connect to a secondary polygon, cut along the short edge to separate it from that polygon. With thick card, carefully score along all the lines so they'll be easier to fold. Assembling it by gluing tabs and folding is fairly intuitive, but we suggest gluing just a few at a time and waiting until the glue dries before continuing. Use paper clips or masking tape – but not a tape that'll damage the card when you pull it off – to hold joints together until the glue dries. Consider using a piece of stiff wire, or a needle, to push the last few joints together from the inside.

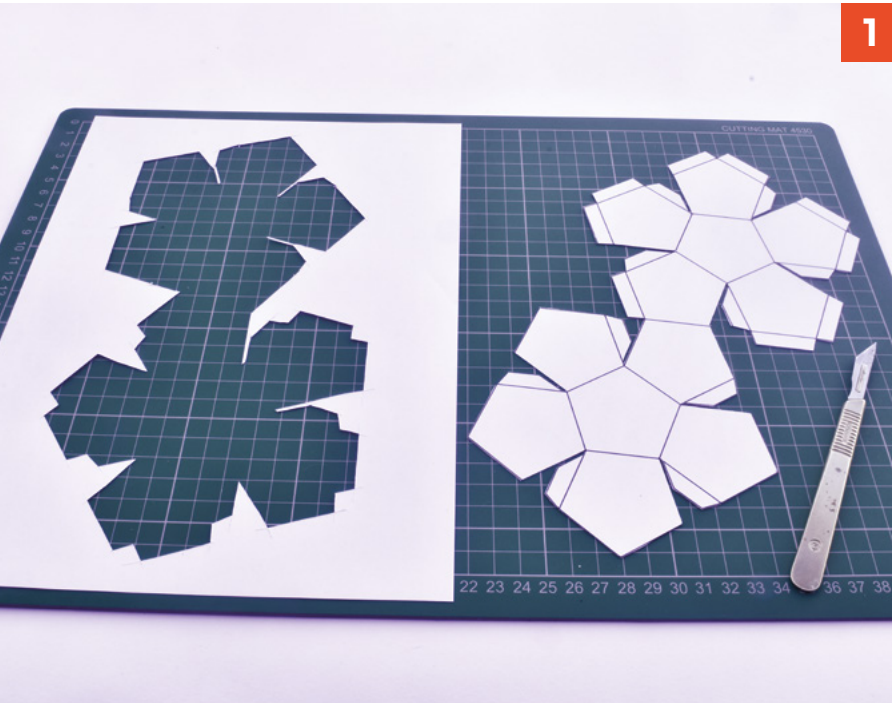
Thick card will make a better polyhedron, but there'll be a limit to the weight of card your printer supports, and if you go above that, you risk damaging the printer. The maximum might only be 200 gsm, which isn't too much thicker than good-quality paper. Our solution was to print two nets onto thinner card. Then we cut out all the individual polygons from one of the nets and glued them onto the other net. As well as overcoming your printer's limitations, it helps with coloured polyhedra. First, it avoids using lots of ink, and second, it'll prevent you smudging the printed, solid coloured areas while you're gluing

and assembling the polyhedron, or subsequently if you handle it with sweaty fingers. We used white card but, in hindsight, if you're pasting on coloured polygons, print the base net onto grey card so that if it shows through at the edges, it won't be too obvious. There are lots of different ways of colouring regular polyhedra so no adjacent faces are the same shade, but just a few colours can look better than lots. You'll see in the photo (overleaf) that we used four colours for our dodecahedron. →

Above ♦ These are the only convex regular polyhedra, the Platonic solids, and their nets. Left to right: tetrahedron, cube, octahedron, dodecahedron, and icosahedron



Left ♦ The Kepler-Poinsot polyhedra have identical regular polygons as their faces, but those faces intersect. Clockwise from top left: small stellated dodecahedron, great stellated dodecahedron, great icosahedron, and great dodecahedron



1

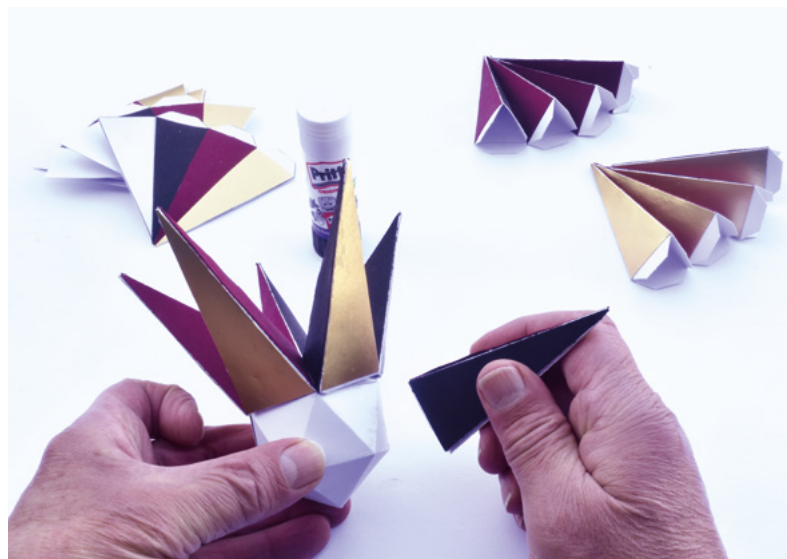


2

THE INS AND OUTS

We described the Platonic solids as convex, so presumably some polyhedra aren't. In fact, if we stick with regular polyhedra, there are another four that aren't convex. They're the Kepler-Poinsot polyhedra: the small stellated dodecahedron, the great stellated dodecahedron, the great icosahedron, and the great dodecahedron. And when we say they're not convex, we mean they have faces that go in and out, so they can be thought of the 3D equivalent of polygons, like stars.

Below ♦ We built a Kepler-Poinsot polyhedron, specifically a great stellated dodecahedron, by fixing 20 triangular pyramids onto the faces of a regular icosahedron



The small stellated dodecahedron looks like an ordinary dodecahedron with five-sided pyramids attached to each of its twelve pentagonal faces. This might suggest why the name includes 'dodecahedron', but the great stellated dodecahedron doesn't fit that theory because it looks like an icosahedron with triangular pyramids attached to each of its 20 triangular faces. Instead, they have the word 'dodecahedron' in their names because they genuinely do have twelve faces, despite them both appearing to have 60 triangular faces. The faces of the Kepler-Poinsot polyhedra are intersecting, as you can see in the images of these

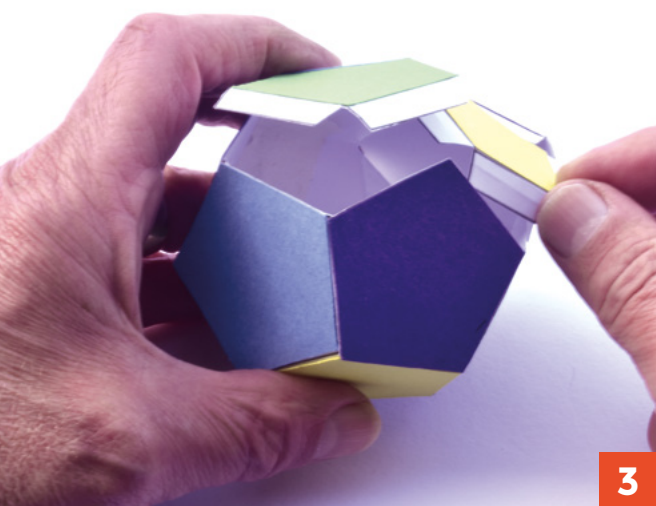
//

Skeletal polyhedra are made of just the edges and, since their faces are missing, you can see right through them

//

strange regular polyhedra (previous page). So, look at the yellow face of the small stellated dodecahedron. Officially called a pentagram, but you can think of it as a star, it's a ten-sided polygon and there are twelve in all – the yellow, red, green, blue, purple, and brown ones you can see, plus six more at the back – so it really is a dodecahedron after all.

You can find nets for the Kepler-Poinsot polyhedra, but we think it's better to make them by gluing



4



add-ons to the base Platonic solids because the end result is stronger.

As you can see in the photo, we made a great stellated dodecahedron by gluing 20 triangular pyramids onto an icosahedron. Nets with the base Platonic solid and the add-ons as separate objects aren't too common, although we found one for the great stellated dodecahedron at hsmag.cc/GreatStellated. Again, we figured that less can be more, so we used an attractive three-coloured design.

A DIFFERENT APPROACH

Even though our models are hollow, polyhedra are really solid objects. But the obviously hollow skeletal polyhedra make an interesting alternative. Skeletal polyhedra are made of just the edges and, since their faces are missing, you can see right through them. They've been made from drinking straws and wire – and you can probably think of other ways to make them – but we recommend an alternative that doesn't differ much from our method of making ordinary card-based polyhedral models.

Put simply, it involves building a polyhedron from a net printed on card but, before folding and gluing it, cutting out the centres from each of the faces. Be careful not to cut away too much of the faces, though, because that would weaken the end result – take a look at the photo (overleaf) for some guidance. An alternative colouring scheme came to mind and the end result looks quite good. Instead →

REAL WORLD POLYHEDRA

Polyhedra aren't just mathematical curiosities. They turn up in nature and have practical applications too.

You're never going to see them unless you have unnatural visual capabilities, but some molecules have their atoms arranged as polyhedra. The atoms form the corners, and the chemical bonds between them form the edges. The most well-known contains 60 carbon atoms and it's called Buckminsterfullerene, or just a Buckyball to its friends. The arrangement of atoms can also cause polyhedra to emerge at much larger scales, and this takes us to minerals, which often exhibit a bewildering variety of polyhedral shapes. For example, pyrite – otherwise known as fool's gold – can have crystals which are cubes, octahedra, pyritohedra which are similar to regular dodecahedra, an octahedron formed as a combination of an octahedron and a pyritohedra, and cubes with some or all of the edges and corners chamfered off, thereby making them as much as 26-hedra. Twinned polyhedra, commonly formed from two intersecting cubes, are also found.



Above

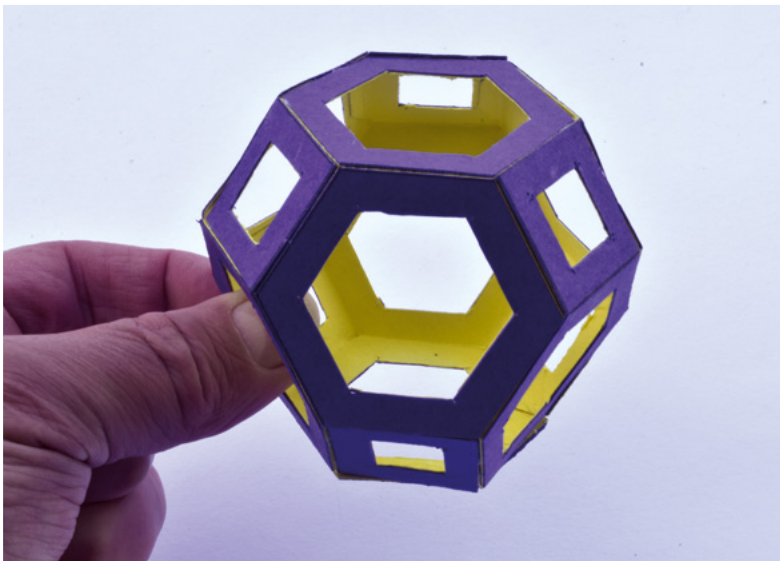
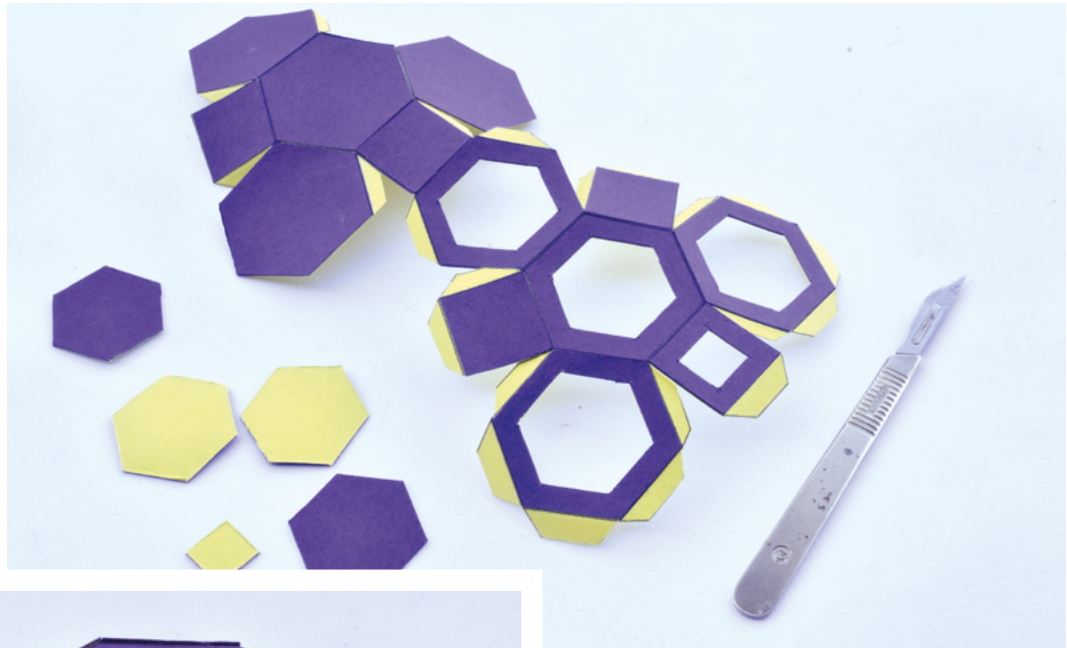
Here's how to make a Platonic solid, specifically a dodecahedron. (1) Print out the net onto thin card and cut it out. (2) Cut out twelve pentagons in your preferred colours of card, and glue them onto the net. (3) Fold the net up, gluing wherever there's a tab. (4) And here's the end result

Right

For a change, we built this uniform polyhedron as a skeletal model. Here you can see the two-coloured net before assembly

Below

The truncated octahedron in all its skeletal glory



of colouring the various faces differently, because you're now going to be able to see the inside, we used just two colours – one for the outside and one for the inside. So, print two copies of the net onto different colours of card and glue them together. Cut off the tabs from the net that'll be on the outside, and separate its individual polygons before gluing, to make the structure easier to fold.

NOT QUITE AS REGULAR

Until now, we've concentrated on regular polyhedra because symmetrical things appeal to our sense of aesthetics. But a degree of non-symmetry can look good too. So, we're going to take a look at some other polyhedra which, while still having some symmetry, are perhaps a bit more interesting than the Platonic solids and the Kepler-Poinsot polyhedra.

What's more, there are a lot more of them, so there's plenty of opportunity to keep you amused for as long as your patience lasts.

Like regular polyhedra, uniform polyhedra also have faces that are regular polygons, but they don't all have to be the same. One class of uniform polyhedra is the Archimedean solids, and there are 13 of them. You can see one that we made in skeleton form – it's called a truncated octahedron and it has twelve faces – six regular hexagons and six squares.



Uniform polyhedra also have faces that are regular polygons, but they don't all have to be the same



Another class of semi-regular polyhedra that's definitely well worth checking out is the polyhedral compound. Put simply, they look like they've been formed from several intersecting polyhedra. The simplest is made from two regular tetrahedra, while the one formed from five tetrahedra is, in our opinion, particularly attractive.

Finally, if you want a real challenge, how about making one of these – or any of the polyhedra we looked at earlier – out of a solid block of wood? Polyhedral models have been made this way, and they can be used as attractive mathematical ornaments, but they'll certainly tax your woodworking skills. □

HACKY RACERS

We race homemade electric go-carts

Come and join us!

» **Fully Charged Live**
29th April - 1st May Farnborough

» **EMF Camp**
2nd-5th June Eastnor, Herefordshire

» **Scrumpy Cup 2**
10th-19th June Cider Farm in Somerset

» **Late Brake Show's Big Brake**
6th-7th August Peterborough

» **Diggerland**
19th-20th August Castleford, Yorkshire

COMPETITORS
& SPECTATORS
WELCOME!



hackyracers



@Hacky_Racers

hackyracers.co.uk



Ian Knight
Z10 Photography

Moving beyond the K40

Blowing smoke



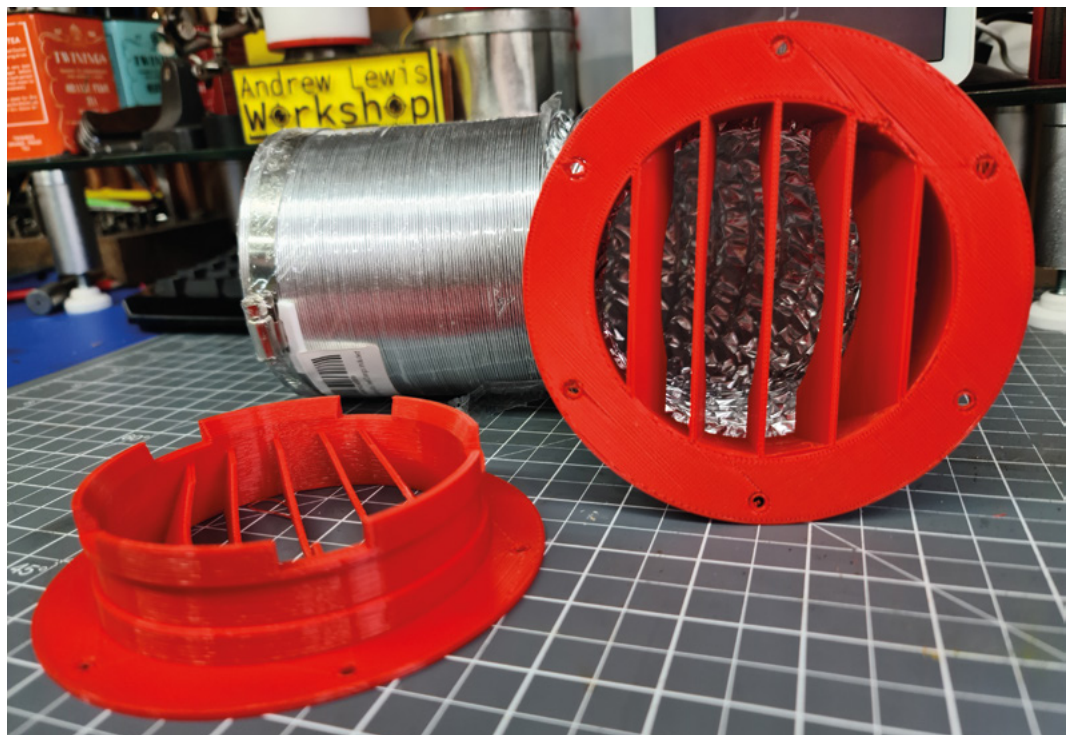
Dr Andrew Lewis

Dr Andrew Lewis is a specialist fabricator and maker, and is the owner of the Andrew Lewis Workshop.

Hitting something with a high-power laser beam doesn't make it magically disappear: it vaporises it, and can also generate a lot of particulates and smoke. Dealing with the vapour cloud from laser cutting isn't as simple as creating a few vent holes in your case. Smoke and gas can prevent the laser from cutting properly and, if left to float around in the laser chamber, they will coat the optics with a layer of opaque residue that causes them to reflect poorly, heat up, and potentially fail. In this article, you'll see how different techniques can help you get rid of unwanted smoke and improve your overall cut quality.

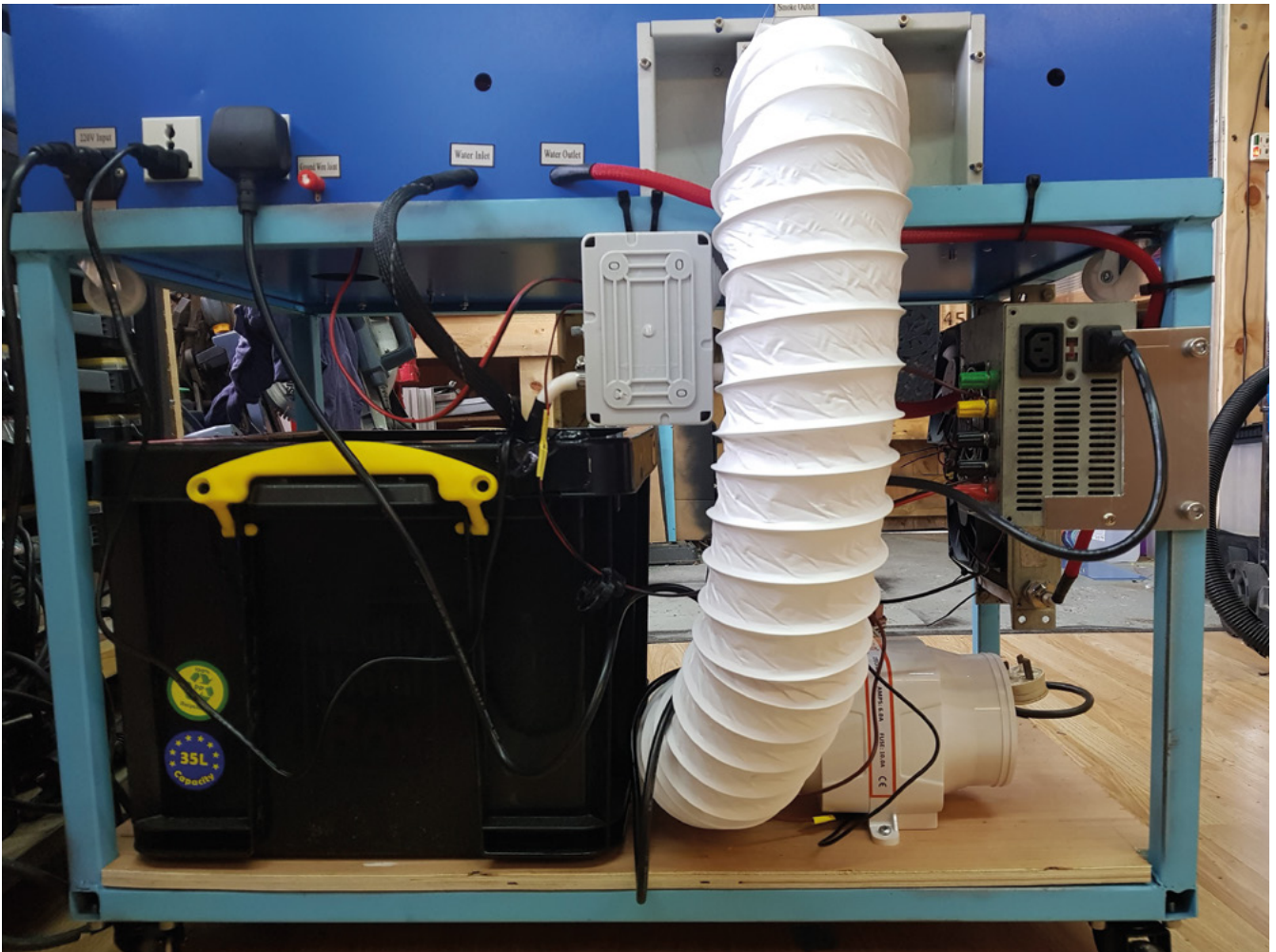
EXTRACTION SYSTEMS SUCK

There are three important types of airflow in a laser cutter, the most obvious of which is extraction. Removing particulates and fumes from the laser chamber, and preventing them from ending up somewhere in the environment, is the most basic level of airflow management. On cheaper laser cutters like the K40, the level of fume extraction included as standard is usually very poor, with a single 120mm fan being used to pull air out of the chamber and dump it directly outside. You can tell that the extraction system is poor, because you will see evidence of smoke or vapour stains on the items you've cut, and you'll smell smoke or plastic in the



Right

With some off-the-shelf components and a 3D printer, you can improve the quality of your laser cutter and keep your lungs safe from harm at the same time



room while you're cutting. Generally speaking, if you can smell the cutter working in an open room, and the extractor fan isn't uncomfortably loud, then fumes are not being extracted effectively.

Even though the purpose of the extractor is simple, there are a few considerations that complicate the implementation of an effective extraction system. Lasers generate heat, and that can lead to sparks, embers, and (in the worst-case scenario) fire being sucked into the extraction system. For that reason, it's not wise to use plastic ducting in areas outside the laser chamber, and definitely avoid using any plastics that release noxious gases if they burn. Most extraction fans aren't designed to deal with hot embers, and it's wise to use an in-line spark arrester before the extractor fan to stop any chance of damage. A spark

arrester is a metal device that deliberately creates turbulence inside a duct to break up the laminar airflow, extinguishing embers before they reach the extractor.

The goal of the extractor fan itself is to clear your laser chamber of smoke as quickly as possible.

There are plenty of options for industrial extraction systems, but they are very expensive to buy. With the proper accessories, a marine bilge blower makes a decent alternative. Bilge blowers are intended to

blow fresh air into the bilge of a boat, pushing any noxious gases out of the bilge vents in the process. They are very effective at extracting fumes from the laser chamber, run from 12V or 24V DC, and can be fitted with a speed controller to optimise the extraction rate (and reduce the noise) to suit the project that you're working on. For most laser →

Definitely avoid using any plastics that release noxious gases if they burn

Above The plastic duct provided with some K40 lasers is a perfect example of what not to use: a flammable ducting that will probably give off dangerous gases if it catches fire

QUICK TIP

Seal up your laser chamber with neoprene tape and silicone to prevent radiation and fumes escaping, but leave deliberate air inlets, or your extractor will try to implode your chamber.

STAY SAFE

A laser cutter is a dangerous machine. Invisible laser radiation can disfigure or blind you, or someone else, permanently. Materials being cut with a laser can release toxic, corrosive, and explosive fumes. Check which materials are safe to work with before you start cutting, and do not cut a material unless you are sure that it is safe to do so. Never operate a laser in an unsafe situation, and never let the machine run out of your control. Always wear laser safety goggles when working with the laser, and be aware of the risks from mechanical parts, high voltages, corrosive gas clouds, unexpected fires, and reflective surfaces. Never leave the laser in a state where it could be activated accidentally by someone who does not understand the risks. Always have appropriate safety equipment nearby.

cutters, a bilge blower rated at about 270CFM (cubic feet per minute) should be enough to keep the chamber clear – that's about three or four times the flow-rate of the fan that comes with some K40 units. It makes sense to filter your exhaust to remove as many pollutants as possible. Venting noxious gases into the environment is very rude, and allowing your laser cutter to do so is similarly uncouth. Gaseous and particulate air pollution have strong negative

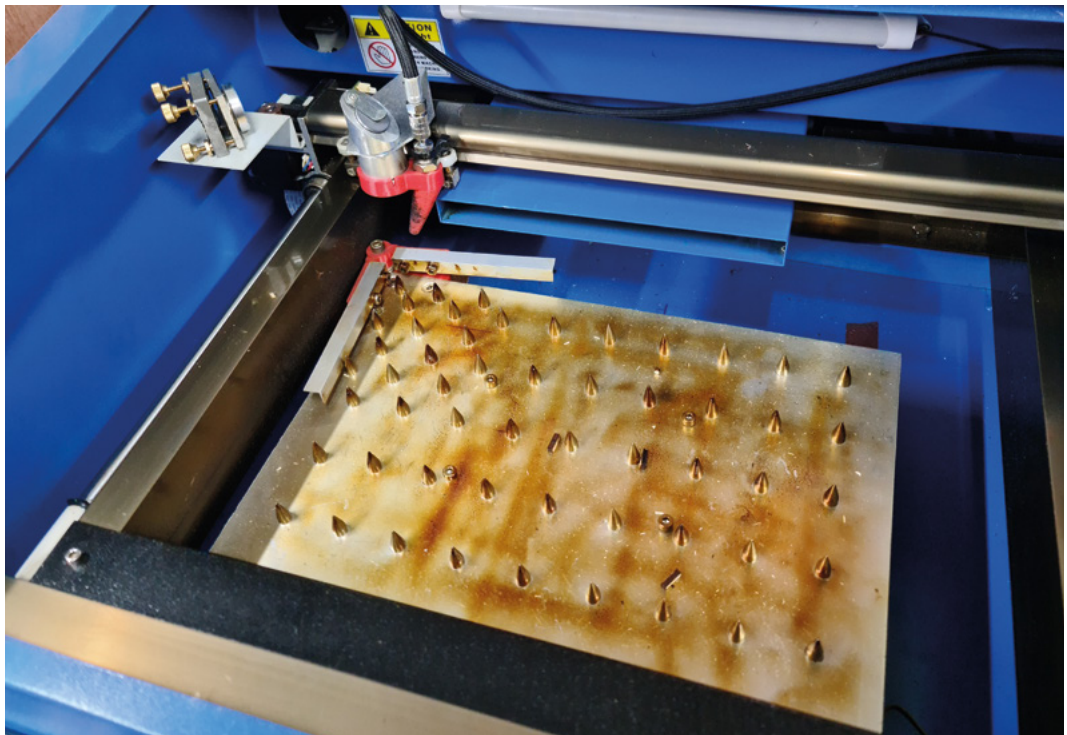
effects on the human body, causing a wide range of symptoms that include dry eyes, a runny nose, and death. It's quite easy to add in-line filters to your exhaust system, as long as you understand that you're filtering two different things from the exhaust. Particulates like smoke and ash can be filtered using coarse and fine particulate filters, similar to those found in vacuum cleaners. Gaseous components of the exhaust can be filtered out using activated carbon. This is the sort of filter you find in a kitchen extractor fan, or an air conditioning system, to remove cooking smells. Arrange the filters in the exhaust ducts so that the coarse filter is closest to the laser cutter, and then place increasingly fine filters towards the exhaust side. How often you change your particulate and carbon filters will depend on how frequently you use the machine, and what type of materials you are working with. If you intend to do a lot of wood or cardboard cutting, you'll need to change your particulate filters more frequently than if you were cutting acrylic or etching glass. As a source of filters, you could consider standard diesel or petrol engine intake filters, vacuum cleaner filters, or pollen filters from an AC system for particulate filtering. You can also buy filter material on the roll and make your own custom filters. Activated carbon filters are available in sheets

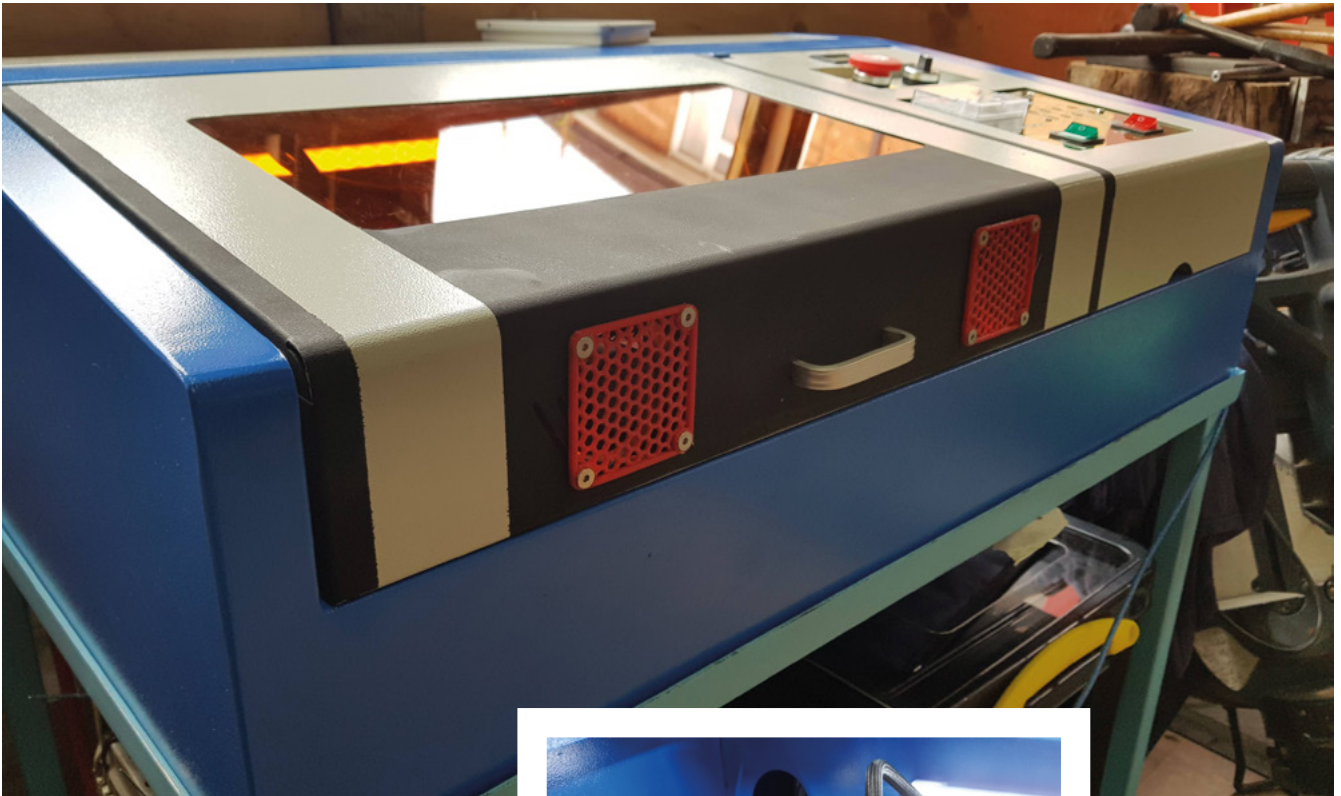
QUICK TIP

Air assist helps make cuts cleaner, but the extra airflow can also keep embers burning and start a fire on some materials. Never leave the laser unattended while it is running.

Right

Both sides of the material will generate smoke, but many systems only concentrate on extracting smoke from above the cut. That can leave nasty residues on your machine bed and on the item you're working with



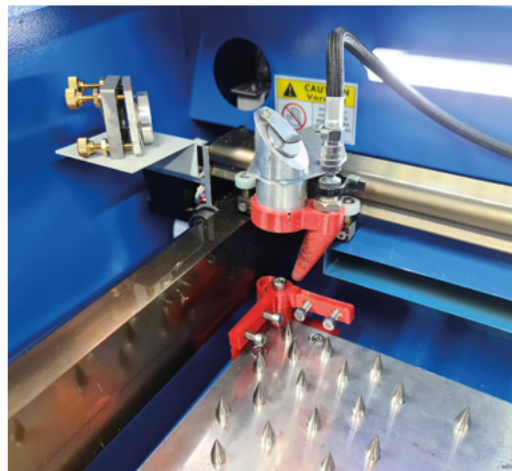


that you can cut to a convenient size. Remember that increasing your filter surface area will improve airflow, so having a large filter box and large filters is better than having discrete in-line filters.

MORE INTAKE THAN EXHAUST MAKES THINGS GET FATTER

Something that hasn't been mentioned here yet is the need for adequate air intakes on your laser cutter. You don't want any unnecessary holes in your laser chamber, because they present an opportunity for a stray laser beam to escape. At the same time, you need to allow adequate ventilation to replenish the air that the extractor is removing, otherwise your machine will start getting thinner when you turn the extractor on. This leads on to the second type of airflow you need to think about, which has been nicknamed 'smoke assist' by some users of the K40.

If the job of the extractor is to remove smoke from the chamber, then the job of smoke assist is to push smoke and gas towards the extractor vent as quickly as possible. Left to its own devices, an extractor will remove smoke from the chamber, but the flow of the smoke towards the vent will eddy in places, and there will be pockets of smoke swirling around, while freshly drawn-in air is vented instead. Smoke assist stops this from happening by making sure the smoke gets pushed out of the chamber by →



Above ♦ These fans have been added to the K40 laser to generate smoke assist. Unfortunately, because of their position in the case, they do very little to move smoke generated below the object. A filter sheet on the inside of the case helps control particulate levels

Left ■ Air assist is an almost essential feature for a laser cutter, particularly if you'll be working with wood. It doesn't matter whether it's built into the lens holder or a 3D-printed add-on, air assist nozzles have a positive effect on the machine optics and the resulting cuts



Above & Below

The difference in quality between parts cut with and without air assist is obvious to see. The wooden enclosure was printed without air assist and looks like it's been smoke-damaged. In contrast, the toy tools have very few smoke marks at all

fans working from multiple angles. The typical setup for smoke assist is with fans mounted in the walls of the laser cutter, with some sort of optical baffles in place to prevent direct laser light from escaping from the chamber. The airflow of the fans should be calculated so that the combined output of the fans is roughly equal to the airflow of the exhaust, although it's better to be slightly above the extraction rate to prevent fumes travelling back into the room where

the laser is situated. It's also wise to include particulate filters on the intakes to prevent any bits of dust or grit getting blown into the chamber from the environment. The filter will also help make sure nothing gets blown out of the chamber if the extractor fails. There are some situations where smoke assist isn't really necessary, and can be a hindrance. If you are cutting something very thin, then the draught from the smoke assist could be enough to blow the part clean off of the bed. Some control boards and firmwares have the ability to control smoke assist and extractor fans in software (Marlin, for example), or you could just use a motor speed controller and a manual switch if that level of remote control is not an option available to you.

An average 120 mm, 12 V PC fan will move 50–70 CFM of air under normal conditions. Silent fans tend to run much less air through, clocking in at about 30–40 CFM. In the context of the laser cutter, silent fans aren't going to make much difference to the overall sound level, but you might choose to use more fans with a lower CFM to control the airflow through the chamber. With a standard bilge blower running at about 270 CFM, you'll be installing somewhere between four and eight fans to match the flow. If you're not sure how things are going,



cover your chamber with a clear plastic sheet, and experiment using some smoke matches or pellets to see how well the extractor is working.

The final type of airflow you need to think about is known as air assist, and it has two main functions. Air assist is the term used to describe a small, directed jet of high-velocity air that is focused more or less directly onto the focus point of the laser. The high-velocity airflow acts as a shield for the laser lens, keeping the area between the lens and the object free from smoke and vapour that would reduce the efficiency of the cutting laser. A secondary effect of the high-velocity air is that the edges of the cut area cool more quickly, and particulates are much less likely to mark the surface of the object you're cutting.

LOUD MEANS WORKING

There are several different ways to implement air assist, and the internet includes a whole spectrum of largely useless or vague suggestions. Many new laser lens mounts have an air assist fitting built into them, which is a good choice if you're happy to buy one. If you're using the K40's stock laser head, then you'll need to come up with an alternative solution. The air assist nozzle is essentially a directional small-bore pipe, so developing a suitable solution isn't difficult. Some users have made their own rigid metal nozzles from brake line, but a 3D-printed nozzle like hsmag.cc/AirAssist works just fine, and is easy to fit onto the head using just a cable tie. Flexible silicone medical tubing, or airbrush line, makes a good lightweight choice for an air hose – the hose isn't under a huge amount of pressure at any time. With a nozzle between 3 and 5mm in diameter, and the highest air velocity around 15mm/s (this is a rough value based on the author's own experience), even a small aquarium air pump producing 15 LPM of air could just about keep up. However, the flow of air from a small aquarium pump isn't ideal because it doesn't produce a constant flow. Smaller pumps often use a single small diaphragm to generate air, and this can cause some 'flutter' in the air supply. For an aquarium this doesn't matter, but when trying to generate air assist, it can have a detrimental effect. More advanced pumps that produce a more powerful output are a better choice. If you have a reasonably sized compressor in your workshop, then

If you're using an air pump, you can simply control the pump using a relay

you could use your shop air to power the air assist. Simply add a valve regulator and water trap to control and condition the flow rate of input air to a usable range. In some cases, the sound of a compressor running for 60 seconds every few minutes is more appealing than having constant air pump noise while the laser runs through a job. If you do choose to use a compressor rather than an air pump, then you can enable or disable the air assist electronically by using a servo-controlled valve between the inlet to the machine and the nozzle of the air assist. If you're using an air pump, you can simply control the pump using a relay. As usual, you

can modify firmware to trigger things as needed using G-code.

With your modified airflow systems, you'll probably notice a few things when you start using your laser cutter. Firstly, the machine will also probably be much

noisier than it was before you started. Extractor fans are loud, air pumps are loud, and even air hissing from a remotely sited compressor can be loud. There isn't a huge amount that you can do about this, without physically positioning blowers and extractors remotely and applying layers of acoustic foam. The noise is a necessary trade-off for the clean air. Noise levels aside, the laser will most likely make much nicer cuts and engravings with less clean-up time needed, particularly on materials like plywood – getting rid of the smoke quickly really helps with that. The environment will also be much less smelly, particularly if you've added carbon and particulate filters to the inlets, as well as the exhaust. ▢

QUICK TIP

Air assist can be both a blessing and a curse, as it can blow bits around the machine like a tiny tornado. Like all machine tools, you'll need to experiment to find the best air speeds and machine feeds for your laser cutter.

AIR SPEED

If you don't know the airflow of a fan, you can calculate it fairly easily if you have access to an anemometer. Airflow is often measured in CFM (Cubic Feet per Minute), sometimes in LPH or LPM (Litres per Hour or Minute), and sometimes m³/s (cubic metres per second). Converting between these units is relatively easy, and there are plenty of online calculators that can do this for you.

Calculating CFM using an anemometer is just a matter of measuring the speed of the air passing through the fan ducting. The maths is simple enough, and if you plug the numbers into a good calculator like hsmag.cc/AirSpeed, you won't even really need to think about it.

EXO-S: developing a swing-wing rocket glider

Is it a bird? Is it a plane?
Is it a rocket?



Jo Hinchliffe

[@concreted0g](#)

Jo Hinchliffe is a constant tinkerer and is passionate about all things DIY space. He loves designing and scratch-building both model and high-power rockets, and releases the designs and components as open-source. He also has a shed full of lathes and milling machines and CNC kit!

Figure 1

Pre-flight photo before the test launch of EXO-S: the EXperimental Open-source Swing-wing Rocket Glider

Living out in rural North Wales, I don't have a local rocketry club, and I only make it to the larger UK rocketry events every now and again as they are all at least a three-hour drive away.

This means I tend to miss some of the

rarer types of rocket projects occasionally flown in the UK. One example is, despite them being quite prevalent in the wider world of model and high-power rocketry, I've never seen a rocket-boosted swing-wing glider in real life.

A swing-wing glider is a fantastically interesting concept, where a rocket motor-propelled glider flies up vertically like a rocket and then changes its configuration to become an aeroplane – it then glides back down to the ground rather than descending under a parachute (**Figure 1**). These work using standard model rocket motors which, as well as acting as a rocket motor, have a small charge that blows out of the other end of the motor after a predetermined delay, usually pushing the nose cone



off and a parachute out. In swing-wing rocket gliders, this charge pushes off the nose cone which is instead connected to a release pin that unfurls the wings.

Armed with only a little research, I wanted to prototype something to see how a wing deployment mechanism might work. I'm a big fan of all types of CAD, including Cardboard Aided Design, so I set about making a card-and-glue prototype. I'd noticed that a lot of the swing-wing designs use an offset release pin mechanism to release the wings, usually because the pin can be run along one side of the fuselage, but I wanted to see how a centre line release pin might work. I drew up some rough shapes in Inkscape and used my budget CNC diode laser rig (reviewed in issue 47 of HackSpace magazine) to cut some cardboard, which could easily have been done by hand with scissors or a craft knife.

I used a couple of sections of bamboo kebab skewer to create some pivot points, and I used a little 0.8mm MIG welding wire to fashion a couple of hooped assemblies which could be held in line with another length of MIG wire acting as a release pin. Small rubber bands were added to the 'wings' which were attached at the other end to a small pin. Pulling the wings together, they could be locked in place with the MIG wire pin and then released to deploy outwards. I added another couple of kebab skewer sections to act as stops to ensure the wings deployed to the correct position. Whilst this cardboard prototype was a messy experiment, it made me consider how a wing stops in position at the end of its travel – it proved to me that a MIG wire pin was strong enough to retain the wings.

Moving to the second prototype, I wanted to 3D-print some kind of fuselage. I also wanted to use balsa wings. Balsa is a great, lightweight material, but I doubted that it was strong enough to withstand the deployment forces, both the pull of the elastic bands and the hard stop in the deployed position. I'd seen that people had reinforced the pivot area of deployable wings using thin plywood, so I planned to do this. I first designed a simple fuselage block in FreeCAD to be 3D-printed. For this second prototype, I wanted to get a correctly sized object with a working release mechanism and wing deployment – I wasn't concerned about it actually flying. Having made a decision about the width of the fuselage and the position of the pivot point, I could then design the

I drew up some rough shapes in Inkscape and used my budget CNC diode laser rig to cut some cardboard

wings and the wing pivots to hopefully fit. At this stage, I started using the idea of two horns on each of the plywood pivot plates. The idea being that when the two wings are folded back and lie over the top of

each other, there are two horns that align in such a way that the release pin can be inserted between them, and that stops them from being pulled open by the rubber bands. The second horn on each pivot plate is positioned so that when

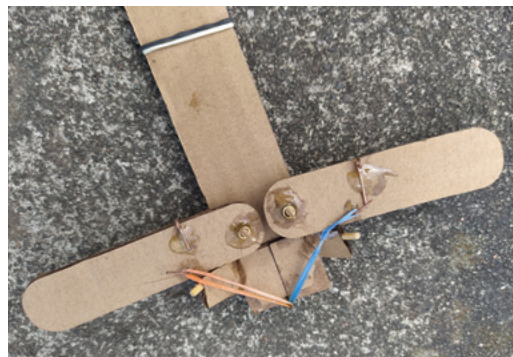
the wings are deployed and the rubber bands pull them into position, they stop the wing as the horn contacts the edge of the fuselage (**Figure 2**, overleaf).

For this prototype, I went with the idea of the pin running along one side of the fuselage, which works →



Above ♦ A close-up of the complete EXO-S wing, fuselage, and motor tube assembly

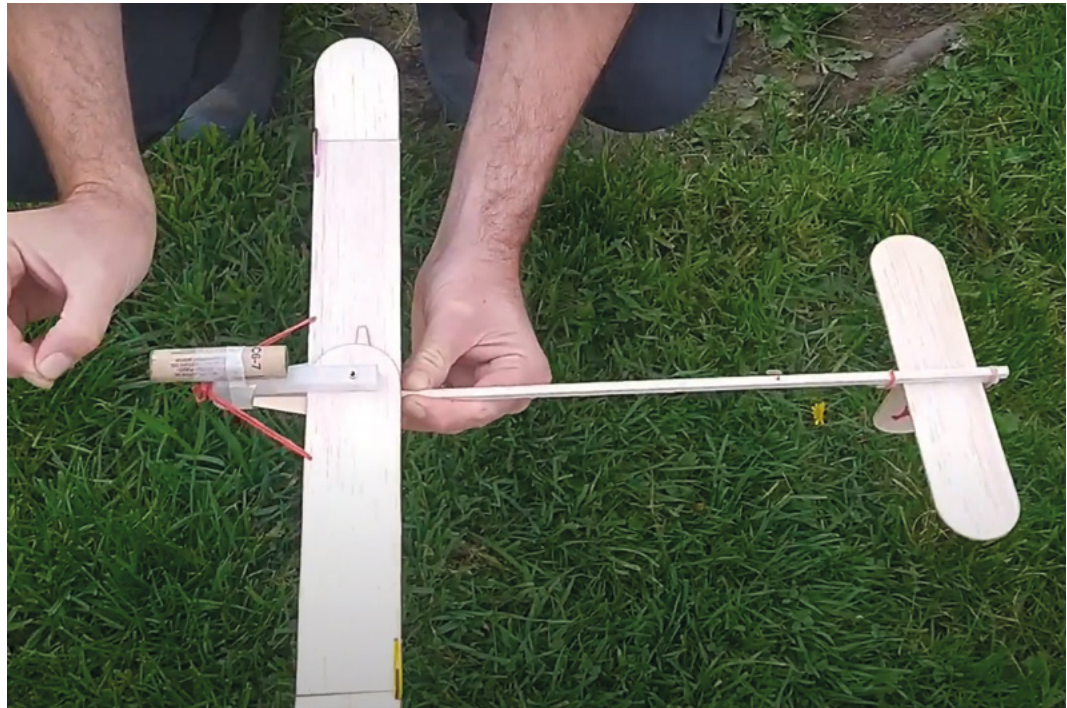
Below ☒ The first cardboard prototype of a pin release wing mechanism



TUTORIAL

Right

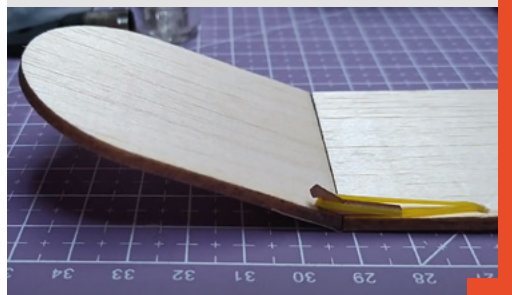
The second prototype allowed me to actually create a working wing deployment system



WING-TIPS

The tips of the wings are hinged and have a rubber band assembly to make them pop up into their final angled position when the wings are deployed. These dihedral tips are useful in that they help create stability in flight and introduce an amount of self-leveling in aircraft. It's also a fun challenge to work out how they can be deployed!

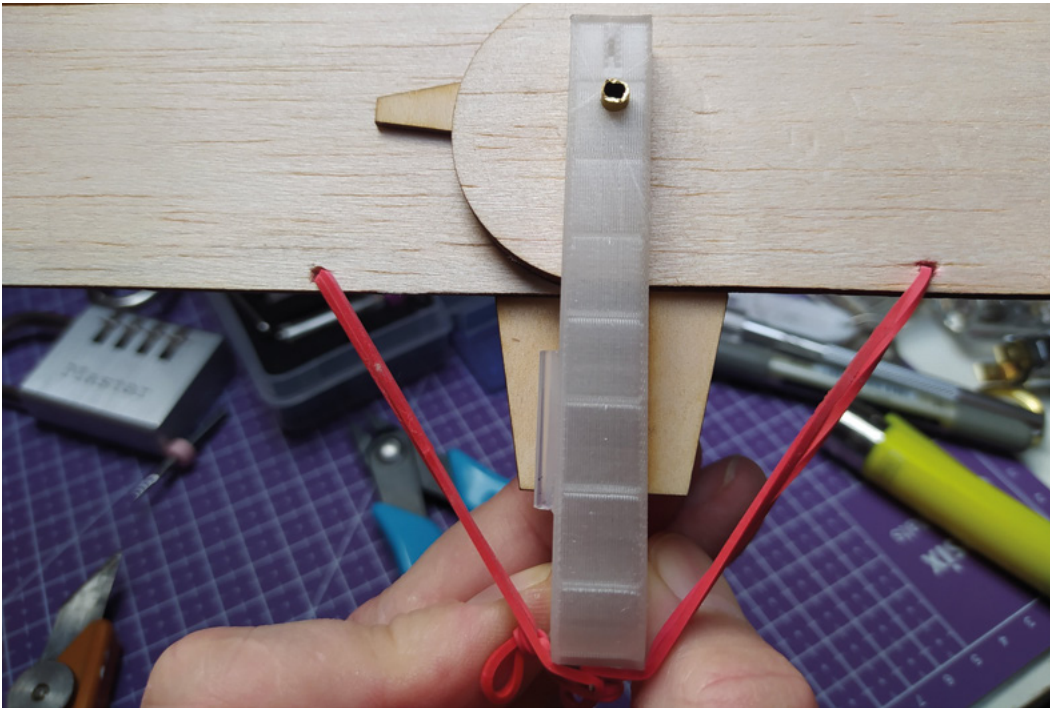
The hinges are made simply from packing tape, and the hinge edges are bevelled to promote the deployed tip sitting at an angle. However, with a rubber band deploying the dihedral tip, they need some form of stop assembly to ensure that they deploy equally. In the first non-flying prototype, I laser-cut some thin plywood which created a hook for the rubber band and an angled stop which limited the movement of the tip. Whilst this worked excellently, it did create quite a large drag-inducing lump at the end of the wings. For the flight prototype, I ended up mounting the rubber band through holes, similar to the wing deployment, and then gluing a small piece of MIG wire bent to an angle onto the wing, which stopped the dihedral tips in the correct position.



fine but does add a little complexity as it makes the design non-symmetric. I drew both the wings and the pivot plates in Inkscape and then laser-cut them. I created a 4 mm diameter pivot point in the 3D-printed fuselage block and used a small piece of brass rod to create the pivot shaft (**Figure 2**). I dimensioned the fuselage so that, hopefully, when the plywood plates and the balsa wings were glued together and stacked, they would fit well with little play in the vertical axis.

At this point, I experimented with attaching rubber bands to the wings. I drilled some 1.5 mm holes through the balsa and decided to reinforce the holes by applying a smear of superglue around the wall of the hole and letting it dry. I then inserted the rubber bands through and experimented with knot tying to try and retain them. This proved to be not a good solution as the knots risked getting trapped between the wings when they were folded into the stowed position, or they could tighten, shrink, and pull through the hole. In the end, I cut some tiny 3-4 mm lengths of the 0.8 mm MIG wire, and after inserting the rubber band through the hole, put these through the rubber band and pulled the system taught. This pulled the MIG wire into the under-surface of the hole and, with the rubber bands under tension, seemed to hold them in place well.

For the release pin holder, I glued a short length of tube (cut from a ballpoint pen ink tube) to the side of the fuselage. I also quickly added a balsa spar and a small tail assembly that I borrowed from another glider I was experimenting with. I added a small MIG wire hook along the spar at the end position of the



wings when stowed – this gently held down the dihedral sections of the wing.

Testing the mechanism revealed a couple of things. The first is that it's incredibly satisfying and an addictive mechanism. Seeing the wings deploy and land in the unfurled position with a satisfying click is a joy and makes the hard work worthwhile. I've made a short video about the EXO-S project with footage of all the test deployments, which are fun to see – you can watch it here: hsmag.cc/EXO-S.

Second, it's quite a violent action, and having filmed the tests, replaying the footage in slow motion revealed a lot of bending and flexing of the balsa spar. Despite knowing this prototype would never fly, I also experimented with taping a used motor to the fuselage to ascertain if I could roughly get the centre of gravity somewhere near the midpoint of the wing, i.e. around the pivot point. It appeared that this would certainly be doable in subsequent prototypes.

I felt after testing that I could probably make the next prototype the first that I might attempt to fly. I had numerous changes I wanted to include based off the earlier prototype. I knew I wanted to make the fuselage body thinner, and I wanted to try and create a fuselage design where the release pin travels through the body rather than featuring on the outside. This has a couple of advantages. One is that, again, the wings and pivot plate design can be symmetrical, and it also potentially reduces drag on the airframe by a small amount. It has to be said, though, that compared to a standard model rocket, these swing-wing gliders are not very streamlined and

create a lot of drag! Finally, I decided that I wanted to make the long spar, out towards the tail section, out of a 4 mm carbon fibre rod to try and minimise the risk of it snapping due to the violence of the wings deploying. I went through a couple of iterations on the two-piece fuselage, leading to this design (**Figure 3**). You can see that the pin is made from MIG welding wire and is placed into its hole before the two sides of the fuselage are brought together.

Having worked out the fuselage body assembly, I then cut the wings and sanded them slightly to create a rudimentary aerofoil from the flat balsa. A better aerofoil would indeed create a more efficient wing with more lift, but it would also create more challenges in designing a deployable wing. Perhaps the next iteration will address this. I designed and cut the horizontal and vertical tail sections, aiming for them to be around 20 percent of the main wing surface area. I also designed and 3D-printed a small →

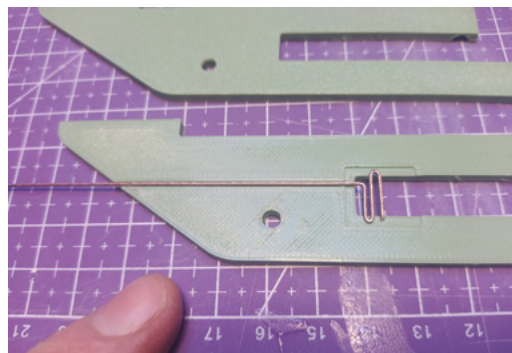


Figure 2  The pivot plates each have two horns: one set forms the release pin catch, and the other set prevents the wings from touching the fuselage body

Figure 3  Using 3D printing meant that creating a fuselage body with internal slots and channels for a release pin was easy to achieve

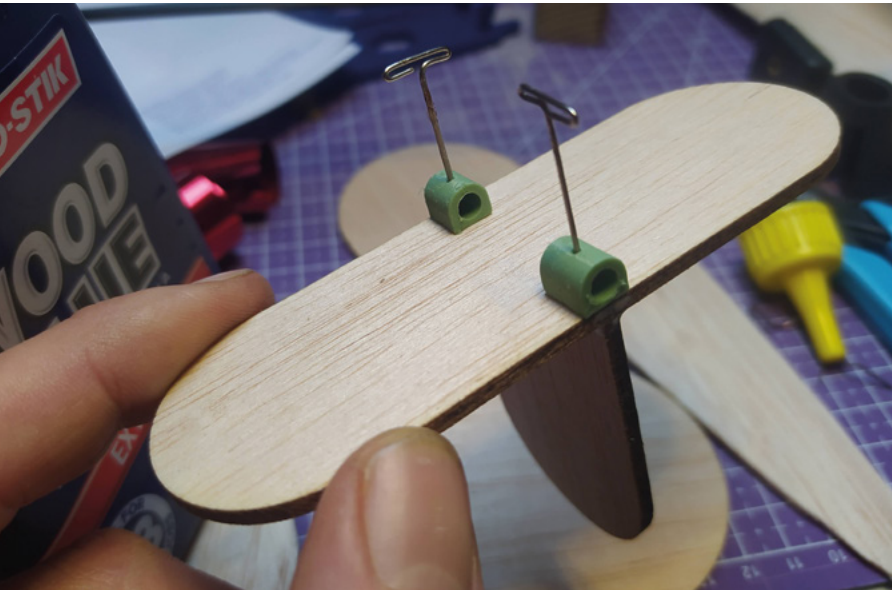


Figure 4 Simple 3D-printed brackets allow the tail components to be glued onto the carbon fibre spar

bracket to fit around the carbon spar, with a flat surface underneath to glue the horizontal tail surface to (**Figure 4**).

With all the pieces cut and printed, I began to assemble the prototype. Turning my attention to the motor mount tube, I cut a length of Estes BT-20 tubing and then 3D-printed a small ring to be glued internally as a thrust ring for the motor to be pushed in against and to transfer the motor thrust to the airframe. I'd modelled the top edge of the fuselage body so that it had a radius that matched the cardboard tubing to maximise the gluing surface. I glued this on after inserting a spent motor and then experimented with the motor tube position to get



Figure 5 The EXO-S aircraft on the launch pad ready for its first flight

the centre of gravity around 25 percent back from the leading edge of the wing. I then cut the release pin wire to length, fashioning it to insert into a slot in the small nose cone I'd designed and 3D-printed using Rocket Workbench in FreeCAD. With the wings attached, pivoting around a short length of the 4 mm carbon tubing, we were ready for a test flight.

Using a spent motor with the wings deployed, I carried out a few test hand launches – gently throwing the airframe over some long grass to see if it glides. It's fair to say that it is on the heavy side and definitely isn't going to float around a lot on thermals, but that said, it does seem to fly the right way up and glide without too much stalling. I was happy for

BALANCING ACT



Back in issue 12, we had a tutorial on using the free and open-source OpenRocket application to help us to design and simulate an example rocket. One of the subjects discussed was the relationship between two points on a rocket's airframe: the centre of gravity (CG) and the centre of pressure (CP). The CG is easy to find, simply finding the point where a rocket or a glider will balance rather than tilt towards the nose or the rear.

The CP is where all the resulting forces acting on a rocket body are centred. You can imagine the CG as the point that a rocket or rocket glider will pivot around if another force acts on it. The CP must be behind the CG and closer to the rear or bottom of the vehicle. When designing a traditional round-tube rocket, designers often aim to get the CP behind the CG at a distance between one and two times the diameter of the body. So, for a 5 cm diameter rocket, the CP might be designed to be 5–10 cm behind the CG. If you imagine the CP is acting as a lever and can create a pivoting force around the CG, then you can further imagine that a longer lever, with the CP further back, can create more movement in the rocket if a force, say a side wind or breeze, pushes the lever. So if you have a CP five times the diameter of the airframe behind the CG, a relatively small breeze may well make the rocket turn near-horizontally into the wind.

With a swing-wing glider, it's harder to simulate stability. However, the EXO-S design and swing wings in this format generally have the CG much closer to the nose cone end than most rockets would – this is due to the position and weight of the motor. The other factor is that in the stowed position for vertical flight, the wings and the tail surface will theoretically pull the CP way back in the airframe, so we automatically have a pretty overstable design – this means they tend to turn into any breeze in the first part of the flight.

this first attempt to simply make the deployment transition, and I wasn't too worried about amazing performance in the deployed section of the flight.

A decent weather day finally arrived, so I ventured to a local rocket flying spot. Setting up the glider on a standard Estes launch pad, I used a second pad and a collection of clothes pegs to help hold the launching wires up at the height they needed to be (**Figure 5**). As the all-up weight was 76 grams, I decided to use a B6-4 motor. With it all set up, I retired to the launch controller box. There was a slight cross-wind, so I was sure that the overstability of the airframe was going to make it turn into the wind. Following a check of the skies and a countdown, the button was pressed. The B6-4 motor presented plenty of thrust, and the EXO-S shot off the pad well. It performed an arcing flight as it predictably turned into the wind. I wondered if it might plummet back to the ground before even deploying. However, with a reassuring 'snick', the deployment charge activated and the wings popped out perfectly – a great transition was achieved. It was a little late, but EXO-S did begin to move to a slightly more horizontal mode and did indeed glide somewhat! I was really pleased with the result. I'd told the few observers that I'd be quite happy if it didn't glide at all but simply performed the deployment of the wings, which it did perfectly.

There was damage to the motor tube as it had landed a little heavily, and whilst I could have straightened it out and reused it, I decided to replace it. I shortened the tube and moved its position to get the centre of gravity closer to the 50 percent point of the wing to try and make the entire airframe less overstable for a more vertical flight path next time. I also replaced the launch lug for a larger diameter one, meaning I could use a thicker and longer launch rail, as, quite obviously, you get less rail remaining with the high up position of the motor with swing-wing gliders.

Below

First launch! It's exciting to see anything you have designed fly for the first time



Above ✦
EXO-S in glide mode during one of the hand-launched test flights

Figure 6 ✦
Replacing and repositioning the motor tube and adding some more colour to try and make the landed craft more visible

I also decorated the new motor tube with some orange vinyl and reprinted a bright orange nose cone as it's difficult to spot the airframe in the brush and grass of the flying field (**Figure 6**). The second flight still arced a little into the wind and was a similar flight to the first with a short glide section at the end – promising results and consistency in the deployment are great to see.

So, I've learnt lots developing this, and I am now totally addicted to ideas around deploying wings! For future developments of the EXO-S project, I have a few redesigns in mind. First up, I need to shed weight to reduce the wing loading; the fuselage body, the pivot plates, and the wings themselves are all target areas to lose a few grams. Secondly, a more refined wing design with more aerofoil and better lift characteristics. I'd also like to play a little with the angles of the wings and tail surfaces to try and create better glide characteristics. It's certainly a fun and challenging set of problems to wrap my grey matter around! You can find the FreeCAD and Inkscape files for the EXO-S on this repository:

hsmag.cc/EXO-S mk1. □



DON'T MISS THE **BRAND NEW** ISSUE!



SUBSCRIBE FROM JUST £5

- **FREE!** 3 issues for the price of one
- **FREE!** Delivery to your door
- **NO OBLIGATION!** Leave any time



**WORTH
£20**

FREE
PI ZERO 2 W*
With your 12-month subscription to the print magazine
magpi.cc/12months

* While stocks last

Buy online: store.rpipress.cc

HackSpace
TECHNOLOGY IN YOUR HANDS

FIELD TEST

HACK | MAKE | BUILD | CREATE

Hacker gear poked, prodded, taken apart, and investigated

PG
110

SPACE SHUTTLE KIT

Learn to solder with
this design classic



PG
112

PRUSA SL1S

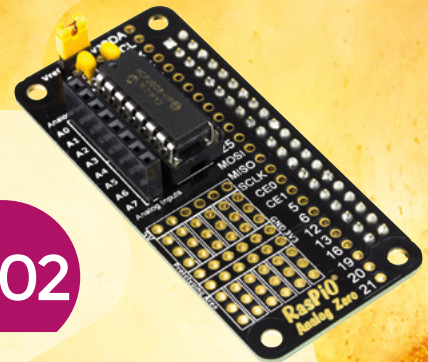
The resin printer
built for speed



PG
102

BEST OF BREED

Music-making for makers



ONLY THE
BEST

Our favourite music-making accessories for your Raspberry Pi

Products to help you make and play music

By Marc de Vinck

 @devinck

Music, whether it be creating, streaming, or just simply distributing it, has always been a popular topic amongst DIY electronics enthusiasts. Although visual-based projects like blinking

LEDs and LCDs certainly take top billing for popularity, many of those projects also involve adding some auditory feedback. And the Raspberry Pi is well-suited to help add some musical flair to your project.

In this roundup, we'll be looking at some audio-related accessories that will have your next Raspberry Pi project singing. The Raspberry Pi, on its own, can perform many different functions for generating sound, recording it, or just playing it back, but there are times when a little extra help would make it much easier. Hopefully these products will inspire you to come up with your own Raspberry Pi-based musical device or, at the very least, get you thinking about your next project!



Piano HAT vs RasPiO Analog Zero

PIMORONI ◆ \$19.22 | pimoroni.com

PIMORONI ◆ \$10.40 | pimoroni.com

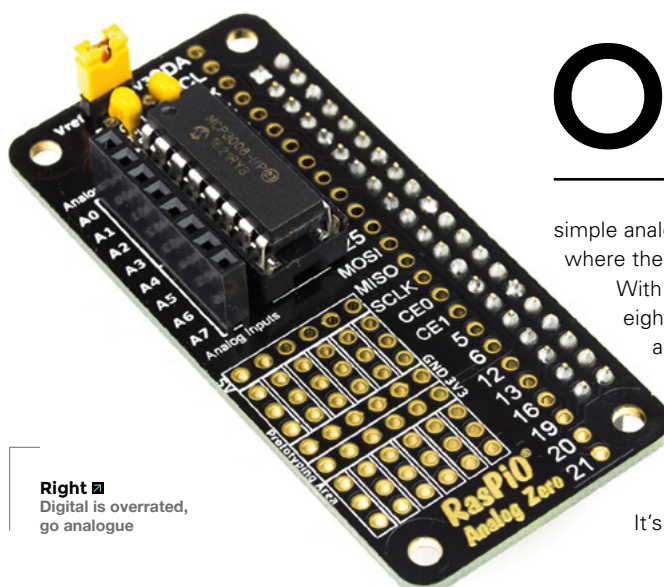
Somewhere between a row of push-buttons on a breadboard and a Moog Matriarch sits the Piano HAT by Pimoroni. It's a beautifully designed and implemented capacitive touch piano HAT for the Raspberry Pi. It

features 16 capacitive touchpads, 13 piano keys, octave up and down buttons, an instrument cycle button, and 16 white LEDs. Make music and a light show!

The Piano HAT is compatible with any 40-pin Raspberry Pi and comes fully assembled and ready to go! Simply plug it in, grab the excellent Python library and example code, and you are ready to use it as a synth controller, make some music, or even use it as an affordable MIDI controller. Head to the Pimoroni website to learn all about this fun little HAT.



Left ■ Tap out your tunes



Right ■ Digital is overrated, go analogue

One thing that the Raspberry Pi isn't particularly good at is reading analogue sensors. It's all digital! And anyone who is looking to create a musical device based on the Raspberry Pi will want a way to read

simple analogue sensors and controls. And that's where the RasPiO Analog Zero comes into play.

With this handy breakout board, you can add eight analogue inputs without any trouble, and the best part – there's nothing else to install. The GPIO Zero Python library is part of Raspberry Pi OS. Simply solder the board together and plug it in. Now your Raspberry Pi can read eight analogue signals simultaneously! It's a synth builder's dream! ➔

VERDICT

Piano HAT

A fun little accessory.

10/10

RasPiO Analog Zero

A simple way of adding analogue in/out.

9/10

Display-O-Tron HAT

PIMORONI ♦ \$28.35 pimoroni.com

know what you are thinking! The Display-O-Tron HAT is not a musical interface. And you're right, it's not. But how often do you need a display when building a streaming music player, or synth? Often! And who wants a simple 16x2 display when you can have a much cooler-looking and more capable screen like the Display-O-Tron?

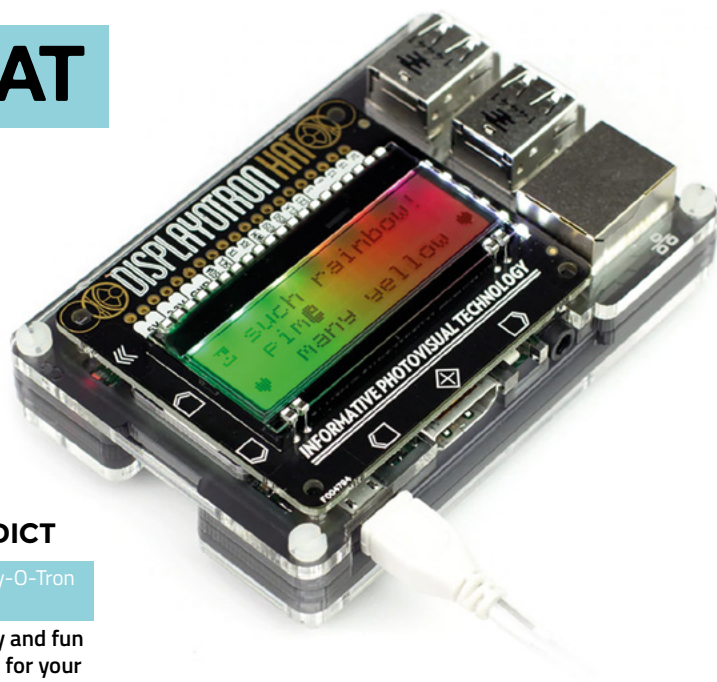
This HAT features six zones of RGB LED back-lighting, a 16x3 LCD display, and six capacitive touch buttons. This board would make a great streaming server display, or IoT music player. Just don't forget, you aren't limited to displaying ASCII characters: you could make some beautiful, albeit low pixel count, graphics too. Take a look at the product page for additional info, and links to the Python library to get you up and running.

VERDICT

Display-O-Tron HAT

A funky and fun display for your Raspberry Pi.

10/10



Left ▣
Add a display to your instrument

Pirate Audio: Headphone Amp

PIMORONI ♦ \$22.68 | pimoroni.com

Pods may be a thing of the past, but the Pirate Audio: Headphone Amp from Pimoroni can make your dream of a DIY portable audio player a reality. Or, you could simply use it as an amp for streaming music from your Raspberry Pi at your desk.

The small-form-factor mini HAT features amplified digital audio via I2S, a dedicated headphone amplifier IC, Low/High gain switch, and a 1.3" IPS colour screen. You also get four tactile buttons, and it comes fully assembled. It's the perfect size for a Raspberry Pi Zero, but it will work with any 40-pin variety of Raspberry Pi. Pimoroni also has detailed instructions on getting started, including a custom Pirate Audio installer that will help configure everything and installs the custom Python library along with the Mopidy extensible music server.



VERDICT

Pirate Audio:
Headphone Amp

Upgrade your
audio.

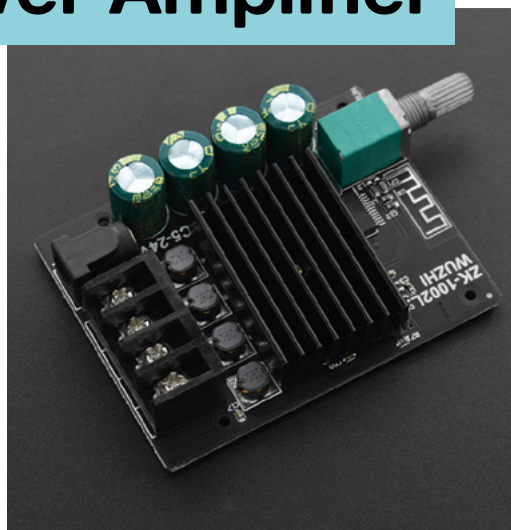
9/10

DFRobot High Power Bluetooth Power Amplifier

SPARKFUN ♦ \$16.95 | sparkfun.com

The DFRobot High Power Bluetooth Power Amplifier Board, available from SparkFun, makes adding a high-power Bluetooth amplifier to your Raspberry Pi simple and at an affordable price. The board features

100-watt output power, a volume control button, and dual-track output. It will work with any flavour Raspberry Pi and a 4–8 ohm speaker from 50–300 watts. Now your Raspberry Pi can transmit two-channel audio over Bluetooth up to 15 metres away! →



Above ☑
Music on the go

Left ♦
For when you need to
blow the roof off

VERDICT

DFRobot High
Power Bluetooth
Power Amplifier

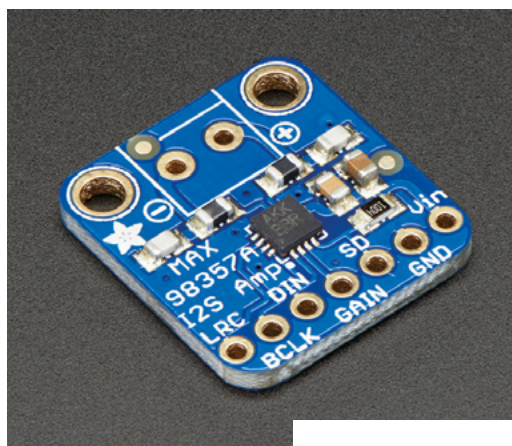
Additional
power for
larger setups.

8/10

Adafruit I2S 3W Class D Amplifier Breakout - MAX98357A

ADAFRUIT ♦ \$5.95 | adafruit.com

If you are considering building an audio player into your next project, then you should take a look at the simple-to-implement Adafruit I2S 3W Class D Amplifier Breakout board. It's a surprisingly powerful little mono amp that's able to send 3.2 watts of power to a 4 ohm impedance speaker. And it works with I2S audio, so there is less noise, making the sound output much cleaner. It also features selectable gain, click and pop reduction, and best of all, it's very affordable. If you want clean sound, grab this board! □

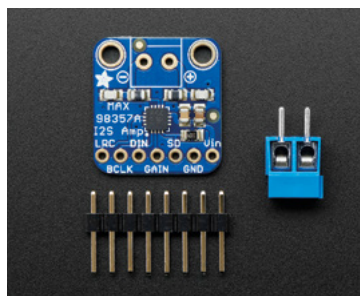
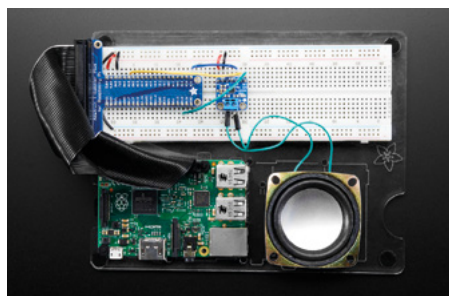


VERDICT

Adafruit I2S 3W Class D Amplifier Breakout - MAX98357A

You can't beat the price of this little amp.

10/10



Above ▣
Amp and analogue output in one

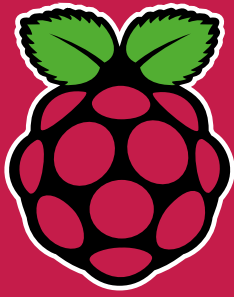
USB AUDIO ADAPTER

ADAFRUIT ♦ \$4.95 | adafruit.com

Yes, the standard Raspberry Pi has an on-board audio jack, but if you want to do some extra processing and potentially get better audio out of your Raspberry Pi, take a look at the USB Audio Adapter from Adafruit. It allows you to plug speakers or headphones directly into your USB port. Adafruit has a complete guide on how to set up your Raspberry Pi to use USB for audio output. It's a simple process. Head on over to the site for complete step-by-step instructions.



THE *Official*

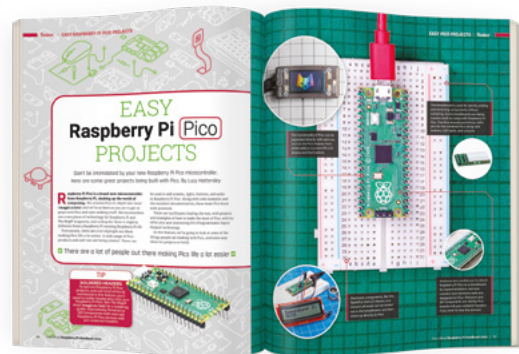


RASPBERRY PI HANDBOOK

2022

200 PAGES OF RASPBERRY PI

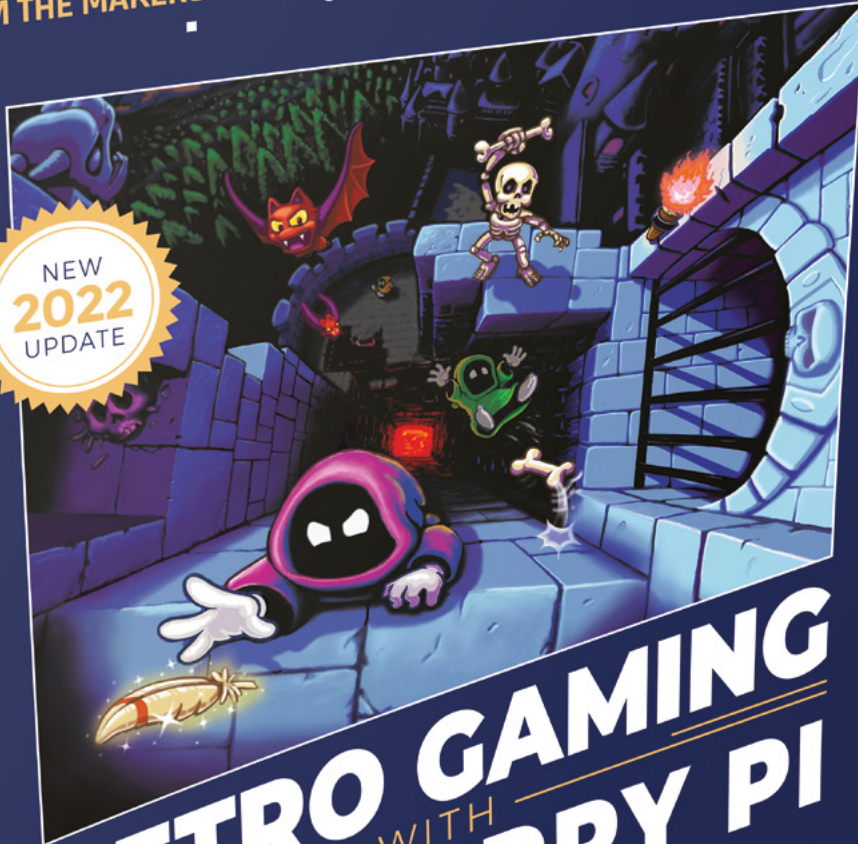
- QuickStart guide to setting up your Raspberry Pi computer
- Updated with Raspberry Pi Pico and all the latest kit
- The very best projects built by your Raspberry Pi community
- Discover incredible kit and tutorials for your projects



Buy online: magpi.cc/store

FROM THE MAKERS OF *The MagPi* THE OFFICIAL RASPBERRY PI MAGAZINE

NEW
2022
UPDATE



RETRO GAMING WITH RASPBERRY PI

2ND EDITION

164 PAGES OF
VIDEO GAME PROJECTS



**PLAY
& CODE
GAMES!**



RETRO GAMING

WITH

RASPBERRY PI

2ND EDITION

Retro Gaming with Raspberry Pi shows you how to set up a Raspberry Pi to play classic games. Build your own games console or full-size arcade cabinet, install emulation software and download classic arcade games with our step-by-step guides. Want to make games? Learn how to code your own with Python and Pygame Zero.

- ***Set up Raspberry Pi for retro gaming***
- *Emulate classic computers and consoles*
- ***Learn to code your own retro-style games***
- *Build a console, handheld, and full-size arcade machine*



BUY ONLINE: magpi.cc/store

Space Shuttle Discovery Soldering Kit

An electronics project that's out of this world

PHYX ♦ £9 | pimoroni.com

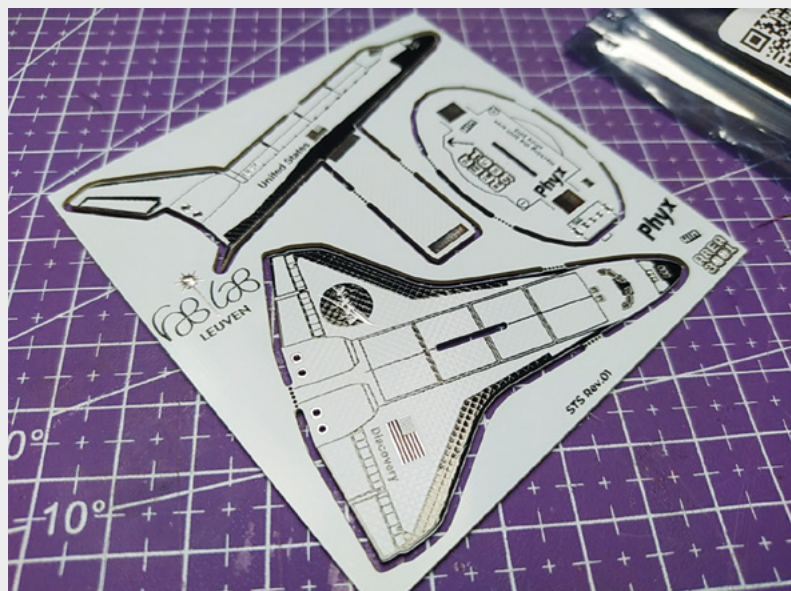
By Jo Hinchliffe

[@concreted0g](https://twitter.com/concreted0g)

Learning to solder is a rite of passage for the aspirant maker, and it's a journey that's well worth undertaking.

Most recommended starting points are small through-hole component kits, where larger components have leads which slide through holes to be soldered – and good target starter projects are often badges, or small electronic experiments which explore a useful electronic design or principle. This kit, designed by Phyx and sold by Pimoroni in the UK, sits at the decorative end of kits but, rather than a badge, it is a small desktop sculpture!

Below ♦
The PCB panel is beautifully designed and is manufactured with a flawless silkscreen

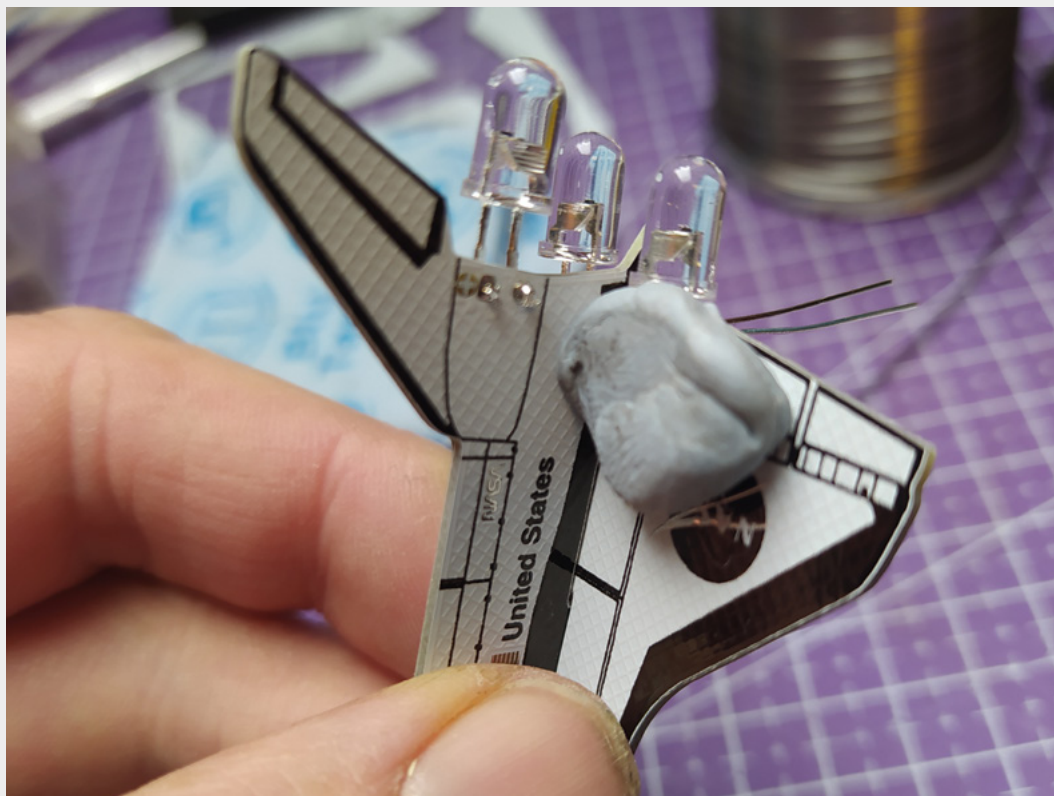


The very neatly designed printed circuit boards (PCBs) slot and solder together to create this free-standing little desk model, based on the NASA Space Shuttle STS Discovery. The base has, on the underside, a coin cell holder (battery not included, but it's a standard CR2032) and a switch. The power is routed through the sculpture to the three flickering LEDs which you mount on the back of the shuttle.

It arrives flat-packed and panelised into one PCB, with the small collection of components heat-sealed into the top section of the anti-static packaging. There are no instructions with the kit, but they are written up as a short list on the Pimoroni product page. It's fair to say that if you have soldered up a few LED badges before tackling this, you probably won't need the instructions anyway as it's pretty intuitive how the boards slot together, and all the positive connections are marked on the exquisite PCB silkscreen so you can work out which way around components go.

The instructions say to solder the LEDs into the boards first, which makes far more sense than what we did, which was to add them after assembling the boards together. We wanted to see what the piece looked like before deciding what length and position to place the LEDs in!

To begin, you carefully de-panelise the PCB by gently bending the PCB sections to break the small perforated tabs that hold them in place. It's optional but, next, you can use a sanding block (or a nail emery board is very good for this) to rub down the remaining bits of the perforated tabs. As we said, probably best to insert the LEDs now if you are new to soldering, or leave them until later if you are confident in your soldering abilities! With our panels ready, we first

**Left** ◆

It's probably simpler to solder the LEDs before assembly but, if not, a blob of Blu Tack can help keep them in place

Below ▣

Once de-panelised, you can optionally sand down the leftover parts of the perforated tabs

soldered the switch to the underside of the base and then the battery holder. Again, this is pretty obvious to place, as the battery holder is clearly outlined in the silkscreen.

With the base components fitted, we set about soldering the two parts that form the shuttle together. When you insert the shuttle upright board, you'll see that there are large rectangular pads on both the upright and the horizontal surface which align at 90 degrees, and you need to solder a bridge across these to create both a conductive joint that is also

a mechanical joint. It's pretty straightforward, and actually suits a slightly wider soldering iron tip, which you probably have if you are a beginner with an entry-level soldering iron. Definitely solder just one side first, and then check the alignment of your panels. It's quite easy to get them not quite straight but, until you solder both sides, it's easy to readjust by remelting the solder joint.

With the shuttle soldered, we repeated the same type of solder connections to solder the bottom of the upright into the base, which again clearly marked the positive side, so you can be sure to get it the right way around. We then set the LEDs in place. Now that the boards receiving the LEDs were assembled, we used a blob of Blu Tack to hold the LED in place whilst we soldered. This is probably a touch harder than doing them first, but it allowed us to look at the LED placement in relation to the whole assembled shuttle!

Once finished, and with a flick of the switch, the LEDs put out a pleasing orange flicker, which gives a great representation of the shuttle motors. One thing that we find with all PCBs that have a white solder mask is that they can look a little bit stained after soldering, and so, we gave ours a quick clean-up using a cotton bud with some isopropyl alcohol on and it was restored to its lovely finish.

We imagine it won't be too long until we buy the other kit in the same series, which is a model of the SR-71 plane! ▣

**VERDICT**

A lovely little desk toy, suitable for beginners to soldering, and offering an alternative starter project to a badge. Excellent.

9/10

Prusa SL1S SPEED

Convert goo into plastic parts as fast as possible

PRUSA ♦ £1678.80 | prusa3d.com

By Alex Bate

🐦 @alexjassic



The latest in Prusa's venture into resin printing, the SL1S SPEED promises high-resolution prints and an unprecedented level of detail, at a whopping three times faster speed than its predecessor, the Prusa SL1.

But does it live up to the hype and, more importantly, the £1679 price point?

"Resin printing is messy and expensive." For years, this is what you'd hear in makerspaces, convention halls, and online forums, as those new to the world of 3D printing questioned whether to invest in a hot plastic or resin printer. On the whole, the argument was valid. Before the days of wash and cure machines, resin printing required a lot of space, time, and patience – not to mention the piles of isopropyl alcohol-filled Tupperware – just to produce anything clean and non-toxic.

Heaps of resin-stained rubber gloves, surrounded by snips, paper towels, and hacked UV curing solutions, littered workbenches and kitchen tables as enthusiastic users reached to open yet another window for ventilation.

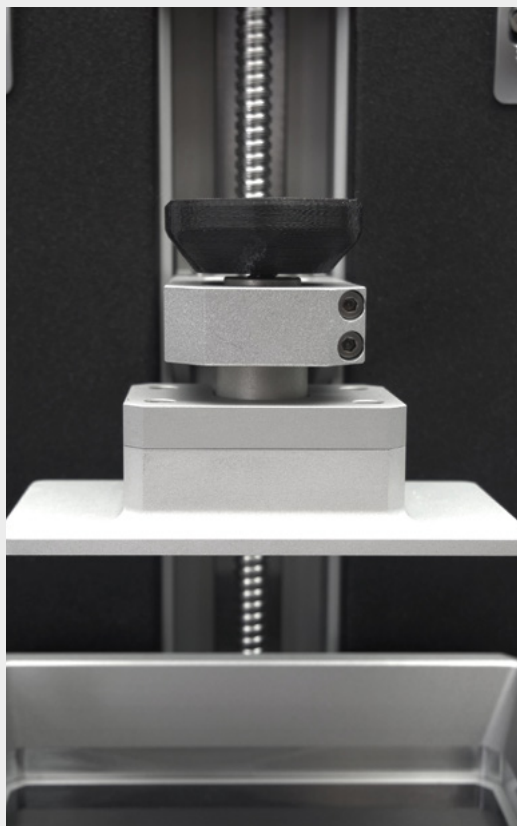
Why would anyone want that when the alternative FDM printer was so neat and tidy in comparison?

But that was then, and this is now, and with the invention of wash and cure machines, fume filters, and tidy desktop enclosures, resin printing has come a long way.

Out of the box, the Prusa SL1S SPEED is a weighty, compact piece of kit with a hard plastic shell and aluminium base that Prusa promises will absorb vibrations for better performance.

Setting up is a breeze as the touchscreen guides you through the process, including collaboration, with the help of clear on-screen instructions and images.

Anyone with prior 3D printing knowledge will no doubt know their way around a number of slicing



Left ♦
The build
plate dips
into the vat
of resin

nine hours less to print four. Now, why you'd want four Benchys is up to you; we won't judge.

But what also came to light during this process was just how easy it was to plug and go with the SL1S. No configuration, no tweaking of settings, we simply pressed a button and off it went, unlike the Ender that took two attempts to get the adhesion down, and another two to unkink some cheap filament.

NEED FOR SPEED?

Resin printing has always outranked FDM when it comes to print detail. It's one of the reasons we've seen a mass increase in resin printer consumption by the model-making and mini-printing communities. FDM simply cannot compete when printing at such a small scale with intricate parts. The SL1S produces highly detailed parts, but so do significantly cheaper competitors.

So, should you spend £1679 on an SLS1 SPEED printer?

Ultimately, and unsurprisingly given the name, it comes down to time. The Prusa prints fast, but it's also quick to set up, easy to use, and – in our tests at least – reliable. All these things save you time which, as any Instagram Business 'Guru' will tell you, is money. How much time you save and how much this is worth to you depends on what you want to print and your own personal circumstances.

If you want minimal-hassle desktop resin printing, then this is a great choice, but it comes with a hefty price tag. ▣

VERDICT

For quality, speed, and ease of use, the SL1S leads the pack, but price is an issue.

9/10

apps, such as Cura and Simplify3D, and will find the transition to resin printing fairly straightforward. For the rest of you, PrusaSlicer, Prusa's open-source tool, is your go-to for prepping your models for printing, and the Prusa YouTube channel, alongside the included beginner's manual, will help you through the process.

Speed: it's in the name so let's talk about it. Prusa promises 1.4 seconds exposure time per layer. Around 2 to 3 seconds per layer is more common for

// The Prusa prints fast, but it's also quick to set up, easy to use, and – in our tests at least – reliable //

desktop resin printers, so the SL1S gives a significant speed boost here.

Whether you're printing one model or multiple, so long as you can fit them all on the 127×80×150mm build volume, your output time doesn't change. Unlike an FDM, where multiple models equal increased print times, resin printers excel at producing prints in mass. To give you an idea of how this speed-up works – compared to the Ender 5 Pro at its highest settings, the SL1S took an hour less to print one Benchy, and



Right ♦
Resin printers
are great at
reproducing
fine detail

issue
#57

ON SALE
21 JULY

RASPBERRY PI PICO

GET THE MOST OUT OF
YOUR MICROCONTROLLER

Also

- EMF CAMP 2022
- GREAT WHITE SHARKS
- BUILD YOUR OWN IOT DEVICE
- GET THE MOST OUT OF YOUR 3D PRINTER
- AND MUCH MORE

DON'T MISS OUT

hsmag.cc/subscribe

next month

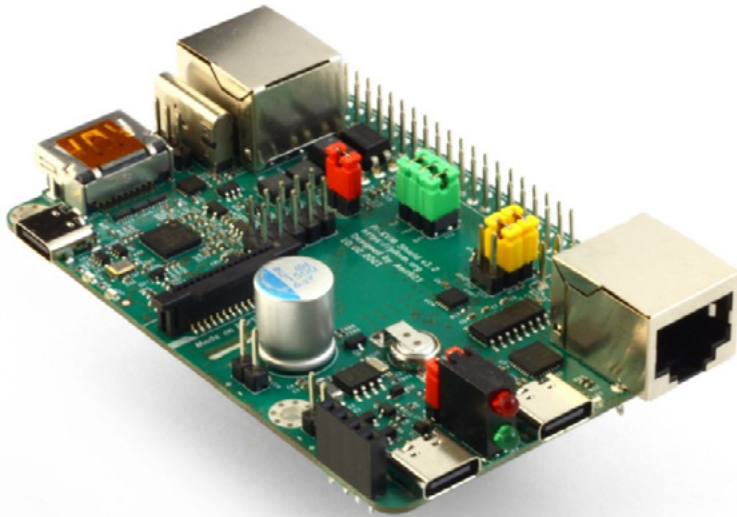


Alex Glow's companion robot F3NR1R, seen here in its cardboard prototype form – if you're iterating through designs of a product, or even just an idea for yourself, cardboard is a cheap, sustainable medium you can use as much of as you want.

PiKVM

Manage your servers or workstations remotely

A **cost-effective** solution for data-centers, IT departments or remote machines!



PiKVM HAT
for DIY and custom projects



Pre-Assembled version

- Real-time clock with rechargeable super capacitor
- OLED Display
- Bootable virtual CD-ROM & flash drive
- Serial console
- Open-source API & integration
- Open-source software

Available at the main Raspberry Pi resellers



Reseller suggestions and inquiries:
wholesale@hipi.io